

## 8-1 Group Functions

### GROUP Functions

- U SQL sledeće funkcije grupe mogu raditi na celoj tabeli ili na posebnoj grupi redova i svaka funkcija vraća jedan rezultat
- Group funkcije: AVG, COUNT, MIN, MAX, SUM, VARIANCE, STDDEV

### GROUP Functions List

- MIN: koristi se sa kolonama koje smeštaju bilo koji tip podataka i vraća minimalnu vrednost
- MAX: koristi se za kolonama koje smeštaju bilo koji tip podataka i vraća maksimalnu vrednost
- SUM: koristi se sa kolonama koje smeštaju numeričke podatke za nalaženje sume vrednosti
- AVG: koristi se sa kolonama koje smeštaju numeričke podatke za računanje srednje vrednosti
- COUNT: vraća broj redova
- VARIANCE: koristi se sa kolonama koje smeštaju numeričke podatke za računanje spread of data around the mean
- Npr, ako je srednja ocena odeljenja 82% na testu a studentski proseci su se kretali od 40% do 100%, variance rezultata će biti veće da su studentski rezultati bili u opsegu od 78% do 88%
- STDDEV: slično kao variance, standardna devijacija meri spread of data
- Npr, za dva seta podataka sa aproksimativno istim mean, što je veći spread, veća je standardna devijacija
- Standardna devijacija i variance mere proširenost podataka oko srednje vrednosti (average ili mean). Bitno je znati da su to grupne funkcije

DEPT_ID	SALARY
90	24000
90	17000
90	17000
60	9000
60	6000
60	4200
50	5800
50	3500
50	3100
50	2600
50	2500
...	...
	7000
10	4400

SELECT MAX(salary)  
FROM employees;

MAX (SALARY)  
24000

### GROUP Functions SELECT Clause

- Funkcije Group su napisane u SELECT izrazu:

```
SELECT column, group_function(column),  
..  
FROM table  
WHERE condition  
GROUP BY column;
```

- Grupne funkcije rade na setu redova da bi dale jedan rezultat po grupi
- WHERE izraz se može uključiti za izvođenje grupne funkcije na sub-setu tabele, npr WHERE department\_id = 90

### GROUP Function Cautions

- Grupne funkcije se ne mogu koristiti u WHERE izrazu:

```
SELECT last_name, first_name  
FROM employees  
WHERE salary = MIN(salary);
```



ORA-00934: group function is not allowed here

## GROUP Function examples

- Primer 1: vraća najmanji broj u life\_expect\_at\_birth kolonu

Examples:	Result
<pre>SELECT MIN(life_expect_at_birth)   AS "Lowest Life Exp" FROM wf_countries;</pre>	32.62
<pre>SELECT MIN(country_name) FROM wf_countries;</pre>	Anguilla
<pre>SELECT MIN(hire_date) FROM employees;</pre>	17-Jun-1987

- Primer 2: koristi kolonu sa znakovnim podacima a vraća zemlju čije ime je prvo u alfabetskom redosledu imena zemalja
- Primer 3: koristi tip podataka datum u koloni i vraća njaraniji hire date

Examples:	Result
<pre>SELECT MAX(life_expect_at_birth)   AS "Highest Life Exp" FROM wf_countries;</pre>	83.51
<pre>SELECT MAX(country_name) FROM wf_countries</pre>	Western Sahara
<pre>SELECT MAX(hire_date) FROM employees;</pre>	29-Jan-2000

Examples:	Result
<pre>SELECT SUM(area) FROM wf_countries WHERE region_id = 29;</pre>	241424
<pre>SELECT SUM(salary) FROM employees WHERE department_id = 90;</pre>	58000

- Može se ograničiti grupa funkcija da bude subset tabele korišćenjem WHERE izraza
- U prethodnoj tabeli primer 1: vraća sumu površina svih zemalja u regionu 29; primer 2: vraća sumu plata za zaposlene u sektoru 90

Examples:	Result
<pre>SELECT AVG(area) FROM wf_countries WHERE region_id = 29;</pre>	9656.96
<pre>SELECT ROUND(AVG(salary), 2) FROM employees WHERE department_id = 90;</pre>	19333.33

- U prethodnoj tabeli, primer 1: srednja vrednost svih površina zemalja u regionu 29; primer 2: vraća srednju platu za zaposlene u sektoru 90, zaokruženu na dve decimale

Examples:	Result
<pre>SELECT ROUND(VARIANCE(life_expect_at_birth), 4) FROM wf_countries;</pre>	143.2394
<pre>SELECT ROUND(STDDEV(life_expect_at_birth), 4) FROM wf_countries;</pre>	11.9683

## GROUP Function and NULL

- Grupne funkcije ignorišu NULL
- U sledećem primeru null vrednosti se ne koriste za traženje srednje vrednosti commission\_pct

```
SELECT AVG(commission_pct)
FROM employees;
```

AVG(COMMISSION_PCT)
.2125

- Tabela employees ima 20 redova; samo 4 zaposlena imaju commission\_pct a ostalih 16 ima NULL. Srednja vrednost se računa traženjem SUM od not-null redova i deljenjem sa COUNT od broja not-null redova

LAST_NAME	COMMISSION_PCT
King	-
Kochhar	-
De Haan	-
Whalen	-
Higgins	-
Gietz	-
Zlotkey	.2
Abel	.3
Taylor	.2
Grant	.15
Mourgos	-
...	...

## More Than One Group Function

- Može se imati više od jedne grupe funkcija u SELECT izraz na istim ili različitim kolonama

```
SELECT MAX(salary), MIN(salary), MIN(employee_id)
FROM employees
WHERE department_id = 60;
```

MAX(SALARY)	MIN(SALARY)	MIN(EMPLOYEE_ID)
9000	4200	103

## 8-2 Count, distinct, NVL

- Grupisanje podataka korišćenjem SQL funkcija omogućava biznisu kompleksno računanje

## COUNT

- COUNT vraća broj non-null vrednosti u koloni izraza (expression column)

```
SELECT COUNT(job_id)
FROM employees;
```

COUNT(JOB_ID)
20

## COUNT and NULL Values

- Dvadeset redova zaposlenih su izlistani u employees tabeli i ako se izabere commission\_pct, dvadeset redova će se vratiti

```
SELECT commission_pct
FROM employees;
```

20 rows returned in 0.01 seconds

- Dodavanje count funkcije u upit COUNT vraća samo četiri
- COUNT specifično broji commission\_pct kolonu ali ignoriše null vrednosti u koloni

```
SELECT COUNT(commission_pct)
FROM employees;
```

COUNT(COMMISSION_PCT)
4

## COUNT All Rows

- COUNT (\*) vraća broj redova iz tabele

- On ne specificira kolonu (koja može ili ne mora imati null) za brojanje; broji broj redova vraćenih u setu rezultata
- Npr, za pronalaženje koliko zaposlenih su zaposleni pre 01/Jan/1996, COUNT se može koristiti u SELECT iskazu

```
SELECT COUNT(*)
FROM employees
WHERE hire_date < '01-Jan-1996';
```

COUNT (*)
9

- Ako je WHERE iskaz uključen u SELECT iskaz, COUNT(\*) vraća broj redova koji zadovoljavaju uslov u WHERE izrazu
- Koristi se COUNT(\*) kada treba biti siguran da brojimo sve redove (uključujući duplikate) kao i one koji mogu imati null u jednij ili više kolona
- Mora se koristiti (\*) pošto sintaksna pravila zahtevaju da svaka funkcija ima najmanje jedan ulazni argument zatvoren u zagradi

### DISTINCT

- Služben areč DISTINCT se koristi za vraćanje samo neduplikatnih vrednosti ili kombinacija neduplikatnih vrednosti u upitu
- Primer: bez korišćenja DISTINCT, upit vraća sve od job\_id vrednosti iz employees tabele, uključujući duplikatne vrednosti

```
SELECT job_id
FROM employees;
```

- Za eliminisanje duplikatnih redova, koristi se DISTINCT službena reč; sada se vraćaju svi job\_id samo jednom, bez duplikatnih

```
SELECT DISTINCT job_id
FROM employees;
```

JOB_ID
AC_ACCOUNT
AC_MGR
AD_ASST
AD_PRES
AD_VP
AD_VP
IT_PROG
...

20 rows returned in 0.01 seconds

vrednosti

12 rows returned in 0.01 seconds

### DISTINCT Non-duplicate

- Služben areč DUPLICATE, kada se koristi u upitu za selekciju više od jedne kolone, vraća non-duplicate kombinacije izabranih kolona
- Primetiti da u primeru ne postoje duplikati kombinacija job\_id i department\_id čak iako duplikati postoje u obe kolone

```
SELECT DISTINCT job_id,
department_id
FROM employees;
```

JOB_ID	DEPARTMENT_ID
IT_PROG	60
SA_REP	80
ST_MAN	50
AD_VP	90
AD_ASST	10
MK_MAN	20
MK_REP	20
SA_MAN	80
SA_REP	-
...	...

13 rows returned in 0.01 seconds

### Using DISTINCT

- Komanda DISTINCT se može koristiti sa svim grupnim funkcijama
- Korišćenje DISTINCT čini da funkcije uzimaju u obzir samo ne-duplikatne vrednosti
- Sledeća dva iskaza daju različite rezultate pošto drugi uzima u obzir samo jednom vrednost 17000

```
SELECT SUM(salary)
FROM employees
WHERE department_id = 90;
```

SALARY	SUM(SALARY)
24000	58000
17000	
17000	
....	

```
SELECT SUM(DISTINCT salary)
FROM employees
WHERE department_id = 90;
```

SALARY	SUM(DISTINCT SALARY)
24000	41000
17000	
17000	
....	

## DISTINCT and COUNT

- Pri korišćenju DISTINCT sa funkcijama grupe kao što je COUNT, set rezultata će vratiti broj vrednosti ne-duplikatnih kolona

```
SELECT COUNT (DISTINCT job_id)
FROM employees;
```

COUNT (DISTINCT job_id)
12

How many different jobs are assigned to employees?

```
SELECT COUNT (DISTINCT salary)
FROM employees;
```

COUNT (DISTINCT salary)
18

How many different salary amounts are paid to employees?

## NVL

- Ponekad je potrebno uključiti null vrednosti u funkcije grupe
- Npr, znajući srednji broj narudžbina kupaca serviranih svakog dana može se saznati koliko hrane treba naručiti svakog meseca
- Takođe vlasnik radnje je shvatio je de bolje uključiti i dane kada restoran ne radi

```
SELECT AVG(NVL(customer_orders, 0))
```

- Another example on employees table:

```
SELECT AVG(commission_pct)
FROM employees;
```

AVG(COMMISSION_PCT)
.2125

```
SELECT AVG(NVL(commission_pct, 0))
FROM employees;
```

AVG(NVL(COMMISSION_PCT,0))
.0425

- Uporediti rezultate sledeća dva upita

```
SELECT AVG(commission_pct)
FROM employees;
```

AVG(COMMISSION_PCT)
.2125

```
SELECT AVG(NVL(commission_pct, 0))
FROM employees;
```

AVG(NVL(COMMISSION_PCT,0))
.0425

- Za te primere se koristi employees tabela sa 20 redova, 4 radnika imaju commission\_pct, ostali null. Srednja vrednost je SUM svih nenull vrednosti a deli se sa COUNT svih nenull redova
- Drugi upit zamenjuje nula za one radnike koji imaju null commission\_pct. Srednja vrednost vraćena se dobija nalaženjem SUM svih 20 redova, i deljenjem sa COUNT od svih 20 redova, pa je srednja vrednost dosta niža