

# **Oracle Academic Initiative**

## **Oracle9i Introduction to SQL**



**Oleh:**

**Tessy Badriyah, SKom.MT**

**Politeknik Elektronika Negeri Surabaya  
Institut Teknologi Sepuluh Nopember  
Surabaya**

## BAB 4 : Fungsi Baris Tunggal

### 4.1. Sasaran

- Memahami bermacam tipe fungsi yang tersedia dalam SQL
- Menggunakan fungsi Karakter, Bilangan dan Tanggal dalam statement SELECT
- Dapat melakukan fungsi Konversi

### 4.2. Fungsi SQL

Fungsi dapat digunakan untuk hal-hal berikut :

- Membentuk kalkulasi pada data
- Memodifikasi item data secara individual
- Manipulasi output dari sekumpulan baris
- Format tanggal dan bilangan untuk keperluan tampilan
- Mengkonversi tipe data kolom

Ada 2 (dua) tipe fungsi :

- Fungsi baris tunggal
- Fungsi baris ganda

### 4.4. Fungsi Baris Tunggal

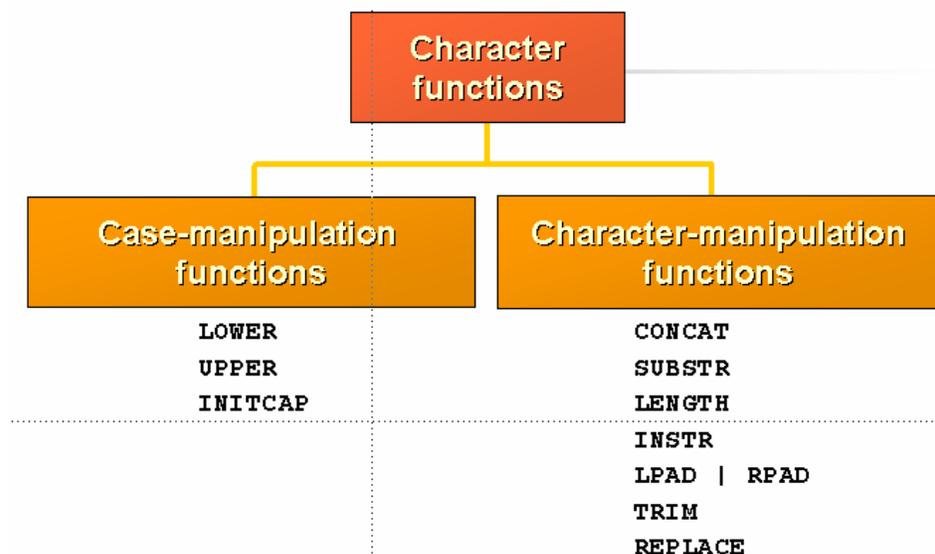
Fungsi baris tunggal beroperasi hanya pada baris tunggal dan mengembalikan satu nilai per baris.

### 4.5. Fungsi Karakter

Sintak penulisan :

*function\_name(column/expression, [arg1, arg2, ...])*

Fungsi karakter menerima karakter sebagai input dan dapat mengembalikan nilai karakter atau bilangan.



Fungsi karakter terbagi menjadi :

- Fungsi konversi, yaitu LOWER, UPPER, INITCAP
- Fungsi manipulasi : CONCAT, SUBSTR, LENGTH, INSTR, LPAD, RPAD, TRIM, REPLACE.

4.6. Manipulasi Fungsi

| Fungsi                  | Hasil      |
|-------------------------|------------|
| CONCAT('Good','String') | GoodString |
| SUBSTR('String',1,3)    | Str        |
| LENGTH('String')        | 6          |
| INSTR('String','r')     | 3          |
| LPAD(sal,10,'*')        | *****5000  |
| RPAD(sal,10,'*')        | 5000*****  |

4.7. Penggunaan dari Manipulasi Fungsi

```

SELECT employee_id, last_name, department_id
FROM employees
WHERE last_name = 'higgins';
no rows selected
    
```

```

SELECT employee_id, last_name, department_id
FROM employees
WHERE LOWER(last_name) = 'higgins';
    
```

| EMPLOYEE_ID | LAST_NAME | DEPARTMENT_ID |
|-------------|-----------|---------------|
| 205         | Higgins   | 110           |

4.8. Fungsi untuk Manipulasi Karakter

| Function                    | Result     |
|-----------------------------|------------|
| CONCAT('Hello', 'World')    | HelloWorld |
| SUBSTR('HelloWorld',1,5)    | Hello      |
| LENGTH('HelloWorld')        | 10         |
| INSTR('HelloWorld', 'W')    | 6          |
| LPAD(salary,10,'*')         | *****24000 |
| RPAD(salary, 10, '*')       | 24000***** |
| TRIM('H' FROM 'HelloWorld') | elloWorld  |

4.9. Menggunakan Fungsi untuk Manipulasi Karakter

```

SELECT employee_id, CONCAT(first name, last_name) NAME,
       job_id, LENGTH (last name),
       INSTR(last_name, 'a') "Contains 'a'?"
FROM employees
WHERE SUBSTR(job_id, 4) = 'REP';
    
```

| EMPLOYEE_ID | NAME           | JOB_ID | LENGTH(LAST_NAME) | Contains 'a'? |
|-------------|----------------|--------|-------------------|---------------|
| 174         | EllenAbel      | SA_REP | 4                 | 0             |
| 176         | JonathonTaylor | SA_REP | 6                 | 2             |
| 178         | KimberelyGrant | SA_REP | 5                 | 3             |
| 202         | PatFay         | MK_REP | 3                 | 2             |

**4.9. Fungsi Bilangan**

Ada 3 (tiga) fungsi bilangan :

- ROUND : Pembulatan desimal ke angka terdekat.
- TRUNC : Memotong nilai desimal ke posisi yang dikehendaki
- MOD : Mendapatkan sisa pembagian

**4.10. Penggunaan Fungsi ROUND**

Contoh penggunaan fungsi Round :

```

SELECT ROUND(45.923, 2), ROUND(45.923, 0),
       ROUND(45.923, -1)
FROM   DUAL;
    
```

| ROUND(45.923,2) | ROUND(45.923,0) | ROUND(45.923,-1) |
|-----------------|-----------------|------------------|
| 45.92           | 46              | 50               |

DUAL adalah dummy tabel yang digunakan untuk menampilkan hasil suatu fungsi atau hasil kalkulasi.

**4.11. Penggunaan Fungsi TRUNC**

Contoh penggunaan fungsi TRUNC :

```

SELECT TRUNC(45.923, 2), TRUNC(45.923),
       TRUNC(45.923, -2)
FROM   DUAL;
    
```

| TRUNC(45.923,2) | TRUNC(45.923) | TRUNC(45.923,-2) |
|-----------------|---------------|------------------|
| 45.92           | 45            | 0                |

**4.12. Penggunaan Fungsi MOD**

Contoh penggunaan fungsi MOD untuk menghitung sisa pembagian nilai gaji setelah dibagi dengan 5000 untuk semua pegawai yang pekerjaannya 'Sales Representative' :

```

SELECT last_name, salary, MOD(salary, 5000)
FROM   employees
WHERE  job_id = 'SA_REP';
    
```

| LAST_NAME | SALARY | MOD(SALARY,5000) |
|-----------|--------|------------------|
| Abel      | 11000  | 1000             |
| Taylor    | 8600   | 3600             |
| Grant     | 7000   | 2000             |

#### 4.13. Bekerja dengan Tanggal

Database akan menyimpan data tanggal sebagai bilangan (number) kalau kita menggunakan operator aritmatik seperti penambahan dan pengurangan.

| Operasi          | Hasil  |
|------------------|--|
| Date + Number    | Date   |
| Date – Number    | Date   |
| Date – Date      | Jumlah Hari (selisih hari antara tanggal pertama dengan kedua) |
| Date + Number/24 | Date (menambahkan sejumlah jam ke data tanggal)                |

#### 4.14. Menggunakan Operator Aritmatika dengan Tanggal

Berikut ini penggunaan Operator Aritmatika pada tipe data Date (tanggal) :

```
SELECT last_name, (SYSDATE-hire_date)/7 AS WEEKS
FROM employees
WHERE department_id = 90;
```

| LAST_NAME | WEEKS      |
|-----------|------------|
| King      | 744.245395 |
| Kochhar   | 626.102538 |
| De Haan   | 453.245395 |

#### 4.15. Fungsi Tanggal

Berikut ini fungsi untuk tanggal :

| Fungsi                                  | Hasil      |
|---|------------|
| MONTHS_BETWEEN('01-SEP-95','11-JAN-94') | 19.6774194 |
| ADD_MONTHS('11-JAN-94',6)               | 11-JUL-94  |
| NEXT_DAY('01-SEP-95','FRIDAY')          | 08-SEP-95  |
| LAST_DAY('01-SEP-95')                   | 30-SEP-95  |

Contoh penggunaan fungsi ROUND dan TRUNC pada data tanggal :

| Assume SYSDATE = '25-JUL-95': |   |           |
|-------------------------------|---|-----------|
| • ROUND (SYSDATE, 'MONTH')    | → | 01-AUG-95 |
| • ROUND (SYSDATE, 'YEAR')     | → | 01-JAN-96 |
| • TRUNC (SYSDATE, 'MONTH')    | → | 01-JUL-95 |
| • TRUNC (SYSDATE, 'YEAR')     | → | 01-JAN-95 |

#### 4.16. Latihan 4, Bagian 1

1. Buat query untuk menampilkan tanggal hari ini dan beri judul 'TANGGAL'

```
TANGGAL
```

```
-----
06-FEB-03
```

2. Tampilkan nomer pegawai, nama, gaji dan gaji setelah dinaikkan 15%

```
SQL> SELECT empno, ename, sal, sal*1.15
2 FROM EMP;
```

```

EMPNO  ENAME          SAL      SAL*1.15
-----
7369 SMITH          800        920
7499 ALLEN        1600       1840
7521 WARD         1250       1437.5
7566 JONES        2975       3421.25
7654 MARTIN      1250       1437.5
.....
```

#### 4.17. Fungsi Konversi

Fungsi Konversi ada dua macam :

- Implicit Data Type Conversion
- Eksplicit Data Type Conversion

#### 4.18. Implicit Data Type Conversion

Fungsi konversi secara implicit adalah fungsi yang secara otomatis melakukan konversi tipe data berikut :

- Dari VARCHAR2 atau CHAR Ke NUMBER
- Dari VARCHAR2 atau CHAR Ke DATE
- Dari NUMBER Ke VARCHAR2
- Dari DATE Ke VARCHAR2

SQL menyediakan 3 (tiga) fungsi untuk mengkonversi suatu nilai dari satu tipe data ke tipe data yang lain.

| Fungsi                       | Tujuan   |
|------------------------------|--|
| TO_CHAR(numer date[, 'fmt']) | Mengkonversi bilangan atau tanggal ke VARCHAR2 dengan format fmt.  |
| TO_NUMBER(char[, 'fmt'])     | Mengkonversi karakter yang berisi digit ke bilangan dengan format fmt.   |
| TO_DATE(char[, 'fmt'])       | Mengkonversi karakter yang merepresentasikan tanggal ke tanggal dengan format fmt, nilai default jika tidak diformat adalah DD-MON-YY. |

#### 4.19. Explicit Data Type Conversion

SQL menyediakan 3 (tiga) fungsi untuk mengkonversi suatu nilai dari satu tipe data ke tipe data yang lain.

| Fungsi                       | Tujuan   |
|------------------------------|--|
| TO_CHAR(numer date[, 'fmt']) | Mengkonversi bilangan atau tanggal ke VARCHAR2 dengan format fmt.      |
| TO_NUMBER(char[, 'fmt'])     | Mengkonversi karakter yang berisi digit ke bilangan dengan format fmt. |
| TO_DATE(char[, 'fmt'])       | Mengkonversi karakter yang merepresentasikan tanggal ke tanggal dengan |

format fmt, nilai default jika tidak diformat adalah DD-MON-YY.

#### 4.20. Penggunaan Fungsi TO\_CHAR dengan tanggal

```
SELECT last_name,
       TO_CHAR(hire_date, 'fmDD Month YYYY')
       AS HIREDATE
FROM   employees;
```

| LAST_NAME | HIREDATE          |
|-----------|-------------------|
| King      | 17 June 1987      |
| Kochhar   | 21 September 1989 |
| De Haan   | 13 January 1993   |
| Hunold    | 3 January 1990    |
| Ernst     | 21 May 1991       |
| Lorentz   | 7 February 1999   |
| Mourgos   | 16 November 1999  |

20 rows selected.

#### 4.21. Penggunaan Fungsi TO\_CHAR dengan tanggal

```
SELECT last_name,
       TO_CHAR(hire_date, 'fmDD Month YYYY')
       AS HIREDATE
FROM   employees;
```

| LAST_NAME | HIREDATE          |
|-----------|-------------------|
| King      | 17 June 1987      |
| Kochhar   | 21 September 1989 |
| De Haan   | 13 January 1993   |
| Hunold    | 3 January 1990    |
| Ernst     | 21 May 1991       |
| Lorentz   | 7 February 1999   |
| Mourgos   | 16 November 1999  |

20 rows selected.

#### 4.22. Penggunaan Fungsi TO\_CHAR dengan bilangan

```
SELECT TO_CHAR(salary, '$99,999.00') SALARY
FROM   employees
WHERE  last_name = 'Ernst';
```

| SALARY     |
|------------|
| \$6,000.00 |

#### 4.23. Penggunaan Fungsi TO\_NUMBER dan TO\_DATE

Untuk mengkonversi string karakter ke bilangan digunakan fungsi TO\_NUMBER.

Untuk mengkonversi string karakter ke tanggal digunakan fungsi TO\_DATE

#### 4.24. RR Date Format

RR Date Format sama dengan element YY hanya saja menggunakan setting century yang berbeda. RR date format lebih baik digunakan daripada format YY dikarenakan nilai century -nya bervariasi tergantung pada dua digit tahun yang ditentukan dan dua digit tahun saat ini. Untuk suatu nilai tanggal RR Date Format dan format YY bisa mengartikan nilai yang berbeda, seperti pada contoh berikut :

| Tahun saat ini | Tanggal yang diberikan | Diartikan oleh RR | Diartikan oleh YY |
|----------------|------------------------|-------------------|-------------------|
| 1994           | 27-OCT-95              | 1995              | 1995              |
| 1994           | 27-OCT-17              | 2017              | 1917              |
| 2001           | 27-OCT-17              | 2017              | 2017              |

#### 4.25. Contoh RR Date Format

Contoh penggunaan RR Date Format, untuk mencari pegawai yang tanggal mulai kerjanya sebelum 1990, gunakan format RR, hasilnya akan sama dengan apakah perintah SQL dijalankan pada tahun 1999 atau sekarang :

```
SELECT last_name, TO_CHAR(hire_date, 'DD-Mon-YYYY')
FROM employees
WHERE hire_date < TO_DATE('01-Jan-90', 'DD-Mon-RR');
```

| LAST_NAME | TO_CHAR(HIR |
|-----------|-------------|
| King      | 17-Jun-1987 |
| Kochhar   | 21-Sep-1989 |
| Whalen    | 17-Sep-1987 |

#### 4.26. Fungsi Bersarang

Fungsi baris tunggal dapat bertingkat pada sembarang level. Fungsi bertingkat atau bersarang (nested functions) dievaluasi mulai pada level terdalam.

Berikut contoh fungsi bersarang :

```
SELECT last_name,
       NVL(TO_CHAR(manager_id), 'No Manager')
FROM employees
WHERE manager_id IS NULL;
```

| LAST_NAME | NVL(TO_CHAR(MANAGER_ID), 'NOMANAGER') |
|-----------|---------------------------------------|
| King      | No Manager                            |

#### 4.27. Fungsi yang menangani nilai NULL

Fungsi berikut ini menangani nilai NULL :

- Fungsi NVL
- Fungsi NVL2
- NULLIF
- COALESCE

#### 4.27. Fungsi NVL

Fungsi NVL digunakan untuk mengkonversi nilai NULL ke nilai yang diinginkan. Tipe data yang digunakan bisa karakter, tanggal dan bilangan.

Antara tipe data yang dikonversi dan tipe data hasil harus sesuai, misal :

- NVL(comm., 0)

- NVL(hiredate, '01-JAN-97')
- NVL(job, 'No Job Yet')

4.28. Penggunaan Fungsi NVL

```

SELECT last_name, salary, NVL(commission_pct, 0),
       (salary*12) + (salary*12*NVL(commission_pct, 0)) AN SAL
FROM employees;
    
```

| LAST_NAME | SALARY | NVL(COMMISSION_PCT,0) | AN SAL |
|-----------|--------|-----------------------|--------|
| King      | 24000  | 0                     | 288000 |
| Kochhar   | 17000  | 0                     | 204000 |
| De Haan   | 17000  | 0                     | 204000 |
| Hunold    | 9000   | 0                     | 108000 |
| Ernst     | 6000   | 0                     | 72000  |
| Lorentz   | 4200   | 0                     | 50400  |
| Mourgos   | 5800   | 0                     | 69600  |
| Rajs      | 3500   | 0                     | 42000  |

...  
20 rows selected.

4.29. Penggunaan Fungsi NVL2

Fungsi NVL2 menguji ekspresi pertama. Jika ekspresi pertamanya tidak NULL, maka fungsi NVL2 akan mengembalikan ekspresi kedua, tapi jika ekspresi pertama NULL, maka akan dikembalikan nilai dari ekspresi ketiga. Berikut ini penggunaan fungsi NVL2 :

```

SELECT last_name, salary, commission_pct,
       NVL2(commission_pct,
            'SAL+COMM', 'SAL') income
FROM employees WHERE department_id IN (50, 80);
    
```

| LAST_NAME | SALARY | COMMISSION_PCT | INCOME   |
|-----------|--------|----------------|----------|
| Zlotkey   | 10500  | .2             | SAL+COMM |
| Abel      | 11000  | .3             | SAL+COMM |
| Taylor    | 8600   | .2             | SAL+COMM |
| Mourgos   | 5800   |                | SAL      |
| Rajs      | 3500   |                | SAL      |
| Davies    | 3100   |                | SAL      |
| Matos     | 2600   |                | SAL      |
| Vargas    | 3500   |                | SAL      |

8 rows selected.

#### 4.30. Penggunaan Fungsi NULLIF

Fungsi NULLIF membandingkan dua ekspresi. Jika keduanya tidak sama maka fungsi akan mengembalikan ekspresi pertama, jika kedua ekspresi sama maka nilai NULL yang akan dikembalikan

```

SELECT first_name, LENGTH(first_name) "expr1",
       last_name,   LENGTH(last_name) "expr2",
       NULLIF(LENGTH(first_name), LENGTH(last_name)) result
FROM   employees;

```

| FIRST_NAME | expr1 | LAST_NAME | expr2 | RESULT |
|------------|-------|-----------|-------|--------|
| Steven     | 6     | King      | 4     | 6      |
| Neena      | 5     | Kochhar   | 7     | 5      |
| Lex        | 3     | De Haan   | 7     | 3      |
| Alexander  | 9     | Hunold    | 6     | 9      |
| Bruce      | 5     | Ernst     | 5     | 5      |
| Diana      | 5     | Lorentz   | 7     | 5      |
| Kevin      | 5     | Mourgos   | 7     | 5      |
| Trenna     | 6     | Rajs      | 4     | 6      |
| Curtis     | 6     | Davies    | 6     | 6      |

...  
20 rows selected.

#### 4.31. Penggunaan Fungsi COALESCE

Fungsi COALESCE mengembalikan nilai ekspresi pertama yang tidak NULL untuk ekspresi-ekspresi yang berada dalam list :

Sintak : COALESCE(expr1, expr2, ... , exprn)

```

SELECT   last name,
         COALESCE(commission_pct, salary, 10) comm
FROM     employees
ORDER BY commission_pct;

```

| LAST_NAME | COMM  |
|-----------|-------|
| Grant     | .15   |
| Zlotkey   | .2    |
| Taylor    | .2    |
| Abel      | .3    |
| King      | 24000 |
| Kochhar   | 17000 |
| De Haan   | 17000 |
| Hunold    | 9000  |

20 rows selected.

Pada contoh tersebut jika `commission_pct` tidak NULL, maka akan dikembalikan nilai tersebut, tapi jika `commission_pct` NULL, maka akan diuji ekspresi `salary`, jika `salary` NULL maka akan dikembalikan nilai tersebut, tapi jika NULL, maka akan dikembalikan nilai di sebelah kanan `salary` yaitu 10.

#### 4.32. Ekspresi Kondisional

Ekspresi Kondisional menggunakan logika IF-THEN-ELSE di dalam SQL Statement.

Digunakan dua cara, yaitu :

- Ekspresi CASE
- Fungsi DECODE

#### 4.33. Ekspresi CASE

Ekspresi CASE mempunyai bentuk umum sebagai berikut :

```
CASE expr WHEN comparison_expr1 THEN return_expr1
      [WHEN comparison_expr2 THEN return_expr2
      WHEN comparison_exprn THEN return_exprn
      ELSE else_expr]
END
```

#### 4.34. Penggunaan Ekspresi CASE

```
SELECT last_name, job_id, salary,
       CASE job_id WHEN 'IT_PROG' THEN 1.10*salary
                   WHEN 'ST_CLERK' THEN 1.15*salary
                   WHEN 'SA_REP' THEN 1.20*salary
       ELSE salary END "REVISED_SALARY"
FROM employees;
```

| LAST_NAME | JOB_ID     | SALARY | REVISED_SALARY |
|-----------|------------|--------|----------------|
| ...       |            |        |                |
| Lorentz   | IT_PROG    | 4200   | 4620           |
| Mourgos   | ST_MAN     | 5800   | 5800           |
| Rajs      | ST_CLERK   | 3500   | 4025           |
| ...       |            |        |                |
| Gietz     | AC_ACCOUNT | 8300   | 8300           |

20 rows selected.

#### 4.35. Fungsi DECODE

Fungsi DECODE menyediakan fasilitas pencocokan seperti yang dikerjakan oleh CASE atau IF-THEN-ELSE.

Sintak (penulisan) fungsi DECODE :

```
DECODE(col/expression, search1, result1
      [, search2, result2, ..., ]
      [, default])
```

## 4.36. Penggunaan Fungsi DECODE

```

SELECT last name, job id, salary,
       DECODE(job_id, 'IT_PROG', 1.10*salary,
                'ST_CLERK', 1.15*salary,
                'SA_REP', 1.20*salary,
                salary)
       REVISSED_SALARY
FROM   employees;

```

| LAST_NAME | JOB_ID     | SALARY | REVISSED_SALARY |
|-----------|------------|--------|-----------------|
| ...       |            |        |                 |
| Lorentz   | IT_PROG    | 4200   | 4620            |
| Mourgos   | ST_MAN     | 5800   | 5800            |
| Rajs      | ST_CLERK   | 3500   | 4025            |
| ...       |            |        |                 |
| Gietz     | AC_ACCOUNT | 8300   | 8300            |

20 rows selected.

## 4.37. Latihan 4, Bagian 2

1. Tampilkan nama pegawai, tanggal mulai bekerja dan tampilan tanggal hari Senin pertama setelah 6 bulan dia bekerja (beri judul REVIEW).

Format untuk tanggal, misal : "Monday, the Twenty-Fourth of May, 1982"

```

ENAME      HIREDATE  REVIEW
-----
SMITH      17-DEC-80 Monday , the Twenty-Second of June , 1981
ALLEN      20-FEB-81 Monday , the Twenty-Fourth of August , 1981
WARD       22-FEB-81 Monday , the Twenty-Fourth of August , 1981
JONES      02-APR-81 Monday , the Fifth of October , 1981
MARTIN     28-SEP-81 Monday , the Twenty-Ninth of March , 1982
BLAKE      01-MAY-81 Monday , the Second of November , 1981 .....

```

2. Untuk tiap pegawai tampilkan nama pegawai, hitung jumlah bulan pegawai tersebut telah bekerja, lakukan pembulatan keatas (round)

```

ENAME      BULAN_BEKERJA
-----
SMITH      266
ALLEN      264
WARD       263
JONES      262
MARTIN     256

```

3. Buat query yang bisa menampilkan informasi dalam satu kolom yang berisi informasi : <nama pegawai> gajinya <gaji> sebulan tapi ingin <3 kali gaji> beri judul kolom "GAJI IMPIAN"

```

GAJI IMPIAN
-----
SMITH gajinya 800 sebulan tapi ingin 2400
ALLEN gajinya 1600 sebulan tapi ingin 4800
WARD gajinya 1250 sebulan tapi ingin 3750
JONES gajinya 2975 sebulan tapi ingin 8925
MARTIN gajinya 1250 sebulan tapi ingin 3750

```

4. Buat query untuk menampilkan nama dan gaji dengan format 15 karakter panjangnya, rata kiri dan kolom yang kosong diisi dengan '\$', beri label "GAJI"

| ENAME | GAJI                           |
|-------|--------------------------------|
| SMITH | \$\$\$\$\$\$\$\$\$\$\$\$\$800  |
| ALLEN | \$\$\$\$\$\$\$\$\$\$\$\$\$1600 |
| WARD  | \$\$\$\$\$\$\$\$\$\$\$\$\$1250 |

5. Buat query untuk menampilkan nama pegawai yang diawali dengan huruf besar dan diikuti dengan semua huruf kecil beri judul kolom 'NAMA', panjang nama pegawai beri judul kolom 'PANJANG', untuk semua pegawai yang namanya diawali dengan J,A, atau M.

| NAMA   | PANJANG |
|--------|---------|
| Allen  | 5       |
| Jones  | 5       |
| Martin | 6       |
| Adams  | 5       |
| James  | 5       |
| Miller | 6       |

6 rows selected.

6. Buat query yang menampilkan nama dan jumlah komisi, jika komisi sama dengan NULL ganti dengan keterangan "Tidak ada Komisi" dan beri judul 'Komisi'.

| ENAME  | KOMISI           |
|--------|------------------|
| SMITH  | Tidak ada komisi |
| ALLEN  | 300              |
| WARD   | 500              |
| JONES  | Tidak ada komisi |
| MARTIN | 1400             |
| BLAKE  | Tidak ada komisi |
| CLARK  | Tidak ada komisi |

7. Buat query yang menampilkan nama pegawai dan jumlah gaji yang ditampilkan dengan symbol '\*', tiap satu '\*' mewakili ratusan dollar.
8. Gunakan fungsi DECODE untuk menampilkan nilai grade semua pegawai berdasarkan pada nilai kolom JOB\_ID, dengan ketentuan sebagai berikut :

| Job           | Grade |
|---------------|-------|
| AD_PRES       | A     |
| ST_MAN        | B     |
| IT_PROG       | C     |
| SA_REP        | D     |
| ST_CLERK      | E     |
| Selain diatas | 0     |