



**JURUSAN TEKNIK INFORMATIKA
FAKULTAS SAINS DAN TEKNOLOGI
UNIVERSITAS SANATA DHARMA YOGYAKARTA**

**MODUL 4
BAHASA QUERY**

Fungsi Karakter, Bilangan dan Tanggal

A. TUJUAN

- Mahasiswa dapat mengetahui dan menggunakan berbagai jenis fungsi yang tersedia di SQL.
- Mahasiswa dapat menggunakan fungsi karakter, bilangan(*number*) dan tanggal (*date*) dalam perintah SELECT

B. LANDASAN TEORI & LANGKAH PRAKTIKUM

Sering kali untuk mendapatkan baris data yang diinginkan, dibutuhkan penggunaan berbagai fungsi baik untuk pengolahan karakter, bilangan maupun tanggal. Modul ini akan menjelaskan berbagai fungsi yang dapat dimanfaatkan untuk memenuhi kebutuhan tersebut yang akan diterapkan dalam perintah SELECT.

B.1. Fungsi

Fungsi merupakan fasilitas yang berguna dalam SQL dan dapat digunakan untuk melakukan hal-hal berikut:

- Melakukan perhitungan data
- Memodifikasi item data secara individu
- Memanipulasi keluaran (*output*) dari kelompok data
- Memformat penampilan *date* dan *number*.
- Mengkonversi tipe data.

Kadang kala fungsi SQL memerlukan argumen dan selalu mengembalikan suatu nilai. Fungsi yang akan dijelaskan pada modul ini merupakan fungsi SQL versi ORACLE.

Terdapat **2 jenis fungsi yaitu**:

- Fungsi *single-row*
- Fungsi *multiple-row*

Fungsi *single-row* beroperasi hanya pada satu baris tunggal dan mengembalikan satu hasil (*result*) per baris. Ada beberapa macam fungsi *single-row*, seperti :

- *Character*
- *Number*
- *Date*
- *Conversion*
- *General*

Fungsi *multiple-row* dapat memanipulasi *group*/kelompok baris untuk menghasilkan satu hasil dari sekelompok baris tersebut. Fungsi ini dikenal dengan fungsi ***group***.

B.2. Fungsi *Single-Row*

Fungsi *single-row* digunakan untuk memanipulasi item data. Fungsi ini menerima satu atau lebih argument dan mengembalikan satu nilai untuk setiap baris yang dikembalikan dari kueri. Argument dapat salah satu dari :

- Konstanta dari user (*user-supplied*)
- Nilai variabel
- Nama kolom
- Ekspresi

Sifat dari fungsi *single-row* :

- Bekerja pada setiap baris yang dikembalikan oleh kueri
- Mengembalikan satu hasil per baris
- Dimungkinkan mengembalikan suatu nilai data dengan tipe data yang berbeda dengan yang direferensikan (argumen)
- Dimungkinkan menggunakan satu atau lebih argumen
- Dapat digunakan dalam klausa SELECT, WHERE dan ORDERBY; dapat juga di *nested*

Sintak:

```
function_name [(arg1, arg2, ...)]
```

Keterangan sintak:

<i>function_name</i>	nama dari fungsi
<i>arg1, arg2</i>	argumen yang digunakan oleh fungsi, yang dapat direpresentasikan dalam sebuah nama kolom atau ekspresi

Fungsi *single-row* yang akan dibahas adalah:

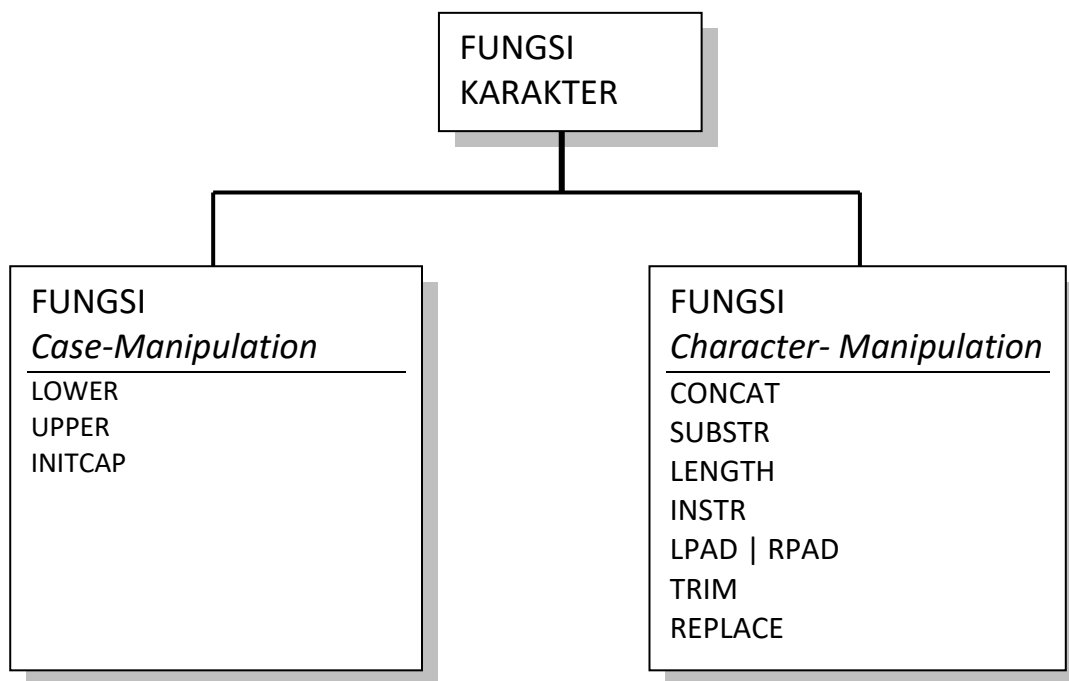
1. Fungsi karakter (*character*) : menerima input karakter dan dapat mengembalikan baik nilai karakter maupun *number*.
2. Fungsi *number* : menerima input numerik dan mengembalikan nilai numerik
3. Fungsi *date* : beroperasi pada nilai tipe data DATE (semua fungsi DATE akan mengembalikan nilai dengan tipe data DATE kecuali fungsi MONTHS_BETWEEN akan mengembalikan nilai dengan tipe data *number*).
4. Fungsi *conversion* : mengkonversi satu nilai dari tipe data satu ke tipe data yang lainnya.
5. Fungsi *general* : NVL, NVL2, NULLIF, COALESCE, CASE, DECODE

B.3. Fungsi Karakter (*Character*)

Fungsi karakter *single-row* menerima data karakter sebagai input dan dapat mengembalikan baik nilai karakter maupun nilai numerik. Fungsi karakter dapat dibagi menjadi 2 kelompok yaitu:

1. Fungsi *case-manipulation*
2. Fungsi *character-manipulation*

Gambar 1, memperlihatkan pembagian fungsi karakter beserta dengan fungsi yang dapat digunakan :



Gambar 1. Pembagian Fungsi Karakter

Tabel 1 adalah tabel fungsi karakter yang berisi sintak fungsi karakter beserta dengan kegunaannya.

Tabel 1. Fungsi karakter dan kegunaannya

FUNGSI	KETERANGAN
LOWER (kolom ekspresi)	Mengkonversi karakter huruf ke huruf kecil
UPPER (kolom ekspresi)	Mengkonversi karakter huruf ke huruf besar
INITCAP (kolom ekspresi)	Mengkonversi karakter huruf ke huruf besar untuk huruf pertama dari setiap kata; huruf yang lain huruf kecil
CONCAT(kolom1 ekspresi1, kolom2 ekspresi2)	Menggabungkan karakter pada kolom 1 dengan karakter pada kolom2
SUBSTR(kolom ekspresi, m, [n])	Mengembalikan karakter-karakter yang diinginkan dari sebuah string, mulai dari karakter posisi m, sepanjang n (Jika m negatif, perhitungan akan dimulai dari akhir string. Jika n dihilangkan, semua karakter sampai akhir string akan dikembalikan)
LENGTH(kolom ekspresi)	Mengembalikan jumlah karakter dalam ekspresi
INSTR(kolom ekspresi, 'string', [m], [n])	Mengembalikan posisi karakter 'string'. Opsional, kita dapat menentukan posisi m untuk memulai pencarian 'string', terjadinya n dari sebuah 'string'. Secara default nilai m dan n adalah 1, artinya akan mencari mulai dari awal dan akan melaporkan kejadian yang pertama.
LPAD(kolom ekspresi, n, 'string')	Membuat string rata kanan dengan mengisi karakter di sebelah kiri dengan 'string' supaya lebar total menjadi n
RPAD(kolom ekspresi, n, 'string')	Membuat string rata kiri dengan mengisi karakter di sebelah kanan dengan 'string' supaya lebar total menjadi n

TRIM(leading trailing both, trim_character FROM trim_source)	Menghilangkan karakter di depan (heading) atau di belakang (trailing) atau keduanya (both) dari sebuah karakter string. Jika <i>trim_character</i> atau <i>trim_source</i> adalah sebuah literal karakter, maka kita harus menutup dengan tanda <i>single quotation</i> (' '). Catatan: kemampuan ini ada sejak Oracle8i.
REPLACE(text, search_string, replacement_string)	Mencari karakter (<i>search_string</i>) dari sebuah text, jika ditemukan, diganti dengan karakter <i>replacement_string</i> .

Selanjutnya adalah contoh-contoh penggunaan fungsi karakter.

1. Fungsi *Case-Manipulation*

Tabel 4.2, memperlihatkan penggunaan fungsi *case-manipulation* beserta dengan contoh.

Tabel 4.2. Contoh Penggunaan fungsi *case-manipulation*.

FUNGSI	HASIL
LOWER ('PRAKTIKUM basisdata')	praktikum basisdata
UPPER ('PRAKTIKUM basisdata')	PRAKTIKUM BASISDATA
INITCAP('PRAKTIKUM basisdata')	Praktikum Basisdata

Berikut contoh-contoh fungsi *case-manipulation* dalam SELECT statement.

Contoh 1:

```
SELECT 'The job id for ' || UPPER(last_name) || ' is '
|| LOWER(JOB_ID) as "EMPLOYEE DETAILS" FROM employees;
```

Hasil :

EMPLOYEE DETAILS
The job id for KING is ad_pres
The job id for KOCHHAR is ad_vp
The job id for DE HAAN is ad_vp
The job id for HUNOLD is it_prog
The job id for ERNST is it_prog
The job id for AUSTIN is it_prog
The job id for PATABALLA is it_prog
The job id for LORENTZ is it_prog
The job id for GREENBERG is fi_mgr
The job id for FAVIET is fi_account
More than 10 rows available. Increase rows selector to view more rows.

Penjelasan:

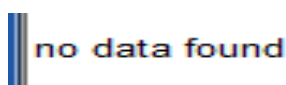
Data yang tersimpan dalam tabel untuk last_name huruf pertama menggunakan huruf besar sedangkan job_id menggunakan huruf besar semuanya (cek dengan perintah `SELECT last_name, job_id from employees;`). Dengan menggunakan fungsi `UPPER(last_name)` maka last_name akan diubah dengan huruf besar seluruhnya dan `LOWER(job_id)` akan mengubah job_id dengan huruf kecil seluruhnya.

Contoh 2:

- Perhatikan kedua contoh berikut yang digunakan untuk menampilkan employee_id, last_name, department_id untuk employee Austin.

```
SELECT employee_id, last_name, department_id
FROM employees
WHERE last_name = 'austin'
```

Hasil :



no data found

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

Hasil :

EMPLOYEE_ID	LAST_NAME	DEPARTMENT_ID
105	Austin	60

1 rows returned in 0,00 seconds [CSV Export](#)

Penjelasan:

Klausula WHERE pada SQL yang pertama, secara spesifik menampilkan employee yang mempunyai last_name austin, akan tetapi karena seluruh data dalam tabel employees disimpan dengan menggunakan huruf yang tepat (case sensitive) maka nama austin tidak ada yang cocok di tabel employees, *no data found*.

Klausula WHERE pada SQL yang kedua menggunakan fungsi LOWER, data yang disimpan pada tabel employees untuk field last_name diubah menjadi huruf kecil seluruhnya untuk tujuan perbandingan. Selama seluruh last_name adalah huruf kecil semua, maka terdapat 1 buah record yang cocok dengan last_name austin.

SQL dapat juga ditulis ulang dengan alasan tertentu yaitu :

```
..... WHERE last_name = 'Austin' .....
```

pencarian dilakukan tepat seperti apa yang disimpan dalam kolom last_name.

Untuk dapat menampilkan last_name dengan huruf pertama adalah huruf besar dapat pula dilakukan dengan menggunakan fungsi UPPER pada perintah SELECT:

```
SELECT employee_id, UPPER(last_name), department_id
FROM employees
WHERE INITCAP(last_name) = 'Austin'
```

2. Fungsi Character-Manipulation

Tabel 4.3 merupakan contoh dan hasil pemakaian fungsi *character-manipulation*

Tabel 4.3. Contoh Penggunaan fungsi *character-manipulation*

FUNGSI	HASIL
CONCAT('Hello','World')	HelloWorld
SUBSTR('HelloWorld',1,5)	Hello
LENGTH('HelloWorld')	10
INSTR('HelloWord','W')	6
LPAD(salary,10,'*')	*****24000
RPAD(salary,10,'*')	24000*****
TRIM('H' from 'HelloWorld')	elloWorld
REPLACE('JACK and JUE', 'J', 'BL')	BLACK and BLUE

Berikut contoh-contoh penggunaan fungsi *character-manipulation* dalam perintah SQL.

Contoh 3

```
SELECT employee_id, CONCAT(first_name, last_name) NAMA,
job_id, LENGTH(last_name), INSTR(last_name, 'a') "Posisi
huruf 'a'" FROM employees WHERE SUBSTR(job_id, 4) = 'REP';
```

Hasil :

EMPLOYEE_ID	NAMA	JOB_ID	LENGTH(LAST_NAME)	Posisi huruf 'a'
1	150 PeterTucker	SA_REP	6	0
2	151 DavidBernstein	SA_REP	9	0
3	152 PeterHall	SA_REP	4	2
4	153 ChristopherOlsen	SA_REP	5	0
5	154 NanetteCambrault	SA_REP	9	2
6	155 OliverTuvault	SA_REP	7	4
7	156 JanetteKing	SA_REP	4	0
8	157 PatrickSully	SA_REP	5	0
9	158 AllanMcEwen	SA_REP	6	0
10	159 LindseySmith	SA_REP	5	0
11	160 LouiseDoran	SA_REP	5	4
12	161 SarathSewall	SA_REP	6	4
13	162 ClaraVishney	SA_REP	7	0
14	163 DanielleGreene	SA_REP	6	0
15	164 MatteaMarvins	SA_REP	7	2

Penjelasan:

Pada Contoh 3, perintah SELECT akan mengembalikan first_name dan last_name yang digabungkan menjadi sebuah alias NAMA, dengan menggunakan fungsi CONCAT, mengembalikan panjang karakter dari last_name dengan fungsi LENGTH,

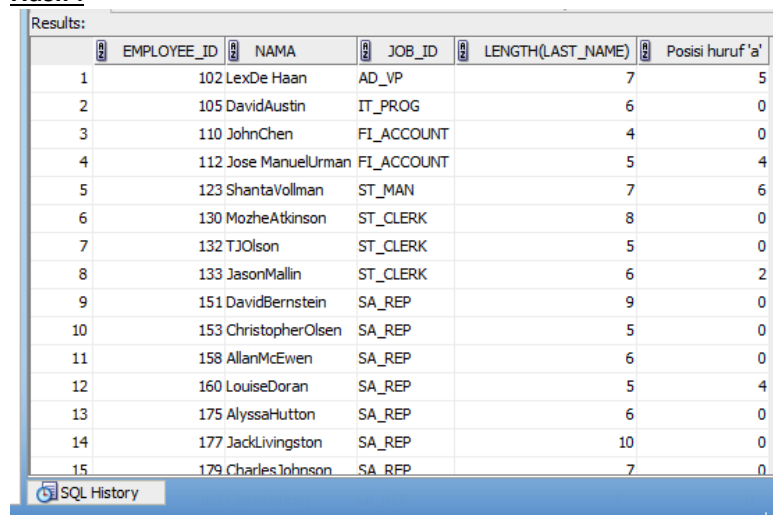
serta mencari posisi huruf a yang pertama dari last_name dengan fungsi INSTR dari seluruh isi tabel employee yang posisi huruf ke-4 dari job_id nya mengandung kata REP.

Contoh 4:

Memodifikasi contoh 3 sehingga employee yang ditampilkan hanyalah yang memiliki huruf terakhir dari last_name nya adalah huruf n.

```
SELECT employee_id, CONCAT(first_name, last_name) NAMA,
job_id, LENGTH(last_name), INSTR(last_name, 'a') "Posisi huruf 'a'"
FROM employees WHERE SUBSTR(last_name, -1, 1) = 'n';
```

Hasil :



	EMPLOYEE_ID	NAMA	JOB_ID	LENGTH(LAST_NAME)	Posisi huruf 'a'
1	102	Lex De Haan	AD_VP	7	5
2	105	David Austin	IT_PROG	6	0
3	110	John Chen	FI_ACCOUNT	4	0
4	112	Jose Manuel Urman	FI_ACCOUNT	5	4
5	123	Shanta Vollman	ST_MAN	7	6
6	130	Mozhe Atkinson	ST_CLERK	8	0
7	132	TJ Olson	ST_CLERK	5	0
8	133	Jason Mallin	ST_CLERK	6	2
9	151	David Bernstein	SA_REP	9	0
10	153	Christopher Olsen	SA_REP	5	0
11	158	Allan McEwen	SA_REP	6	0
12	160	Louise Doran	SA_REP	5	4
13	175	Alyssa Hutton	SA_REP	6	0
14	177	Jack Livingston	SA_REP	10	0
15	179	Charles Johnson	SA_REP	7	0

B.4. Fungsi Number

Fungsi number menerima input numerik dan mengembalikan nilai numerik. Fungsi numerik yang akan dibahas adalah ROUND, TRUNC dan MOD. Tabel 4.4 merupakan tabel daftar fungsi number beserta dengan contoh dan kegunaannya.

Tabel 4.4. Fungsi numerik, kegunaan dan contohnya

FUNGSI	KEGUNAAN	CONTOH FUNGSI	HASIL CTH
ROUND(kolom ekspresi,n)	Pembulatan nilai ke n tempat desimal atau jika nilai n tidak diberikan maka tidak menggunakan tempat desimal. (Jika n negatif, maka pembulatan dimulai dari titik desimal ke kiri sebanyak n tempat desimal)	ROUND(45.926,2)	45.93
TRUNC(kolom ekspresi,n)	Pemotongan nilai ke n desimal. Jika n dihilangkan maka default n = 0	TRUNC(45.926,2)	45.92
MOD(m,n)	Mengembalikan sisa pembagian m dengan n	MOD(1600,300)	100

Berikut contoh-contoh penggunaan fungsi numerik dalam SELECT statement.

Contoh 5

- Menampilkan hasil ROUND suatu bilangan dalam perintah SELECT.

```
SELECT ROUND (45.925, 2) , ROUND (45.926, 0) , ROUND (45.926, -1)
FROM DUAL;
```

Hasil:

Results	Explain	Describe	Saved SQL	History
ROUND(45.926,2) ROUND(45.926,0) ROUND(45.926,-1)				
45,93 46 50				
1 rows returned in 0,00 seconds CSV Export				
Language: id				

Penjelasan:

Round(45.926,2) → pembulatan 2 angka desimal, karena angka ke 3 lebih dari dari 5 yaitu 6 maka angka desimal ke 2 akan dibulatkan menjadi 3 sehingga hasilnya adalah 45.93

Round(45.926,0) → pembulatan 0 artinya tidak ada angka desimal. Karena angka ke 1 digit desimal adalah 9 maka satuan dibulatkan ke atas yaitu menjadi 46.

Round(45.926,-1) → karena n=-1, maka pembulatan dilakukan dari titik desimal ke kiri sebanyak 1 tempat angka desimal. Angka 1 digit di kiri titik desimal adalah 5, karena dibulatkan maka angka 4 menjadi 5 sehingga hasilnya adalah 50.

Catatan : DUAL adalah tabel dummy yang dapat digunakan untuk mengembalikan hasil dari fungsi dan perhitungan

Contoh 6

- Menampilkan hasil TRUNC suatu bilangan dalam perintah SELECT

```
SELECT TRUNC (45.926, 2) , TRUNC (45.926, 0) , TRUNC (45.926, -1)
FROM DUAL;
```

Hasil:

```
TRUNC (45.926, 2)    TRUNC (45.926, 0)    TRUNC (45.926, -1)
-----
                    45.92                      45                      40
```

Penjelasan:

Argumen 2 → memotong 2 digit di belakang koma sehingga menjadi 45.92

Argumen 0 → memotong 0 digit atau sama artinya dengan tidak ada digit dibelakang koma sehingga menjadi 45 dan

Argumen - 1 → memotong 1 digit desimal disebelah kiri titik sehingga menjadi 40.

Contoh 7:

- Menghitung sisa salary dari karyawan yang memiliki job_id SA_REP dengan 5000.

```
SELECT LAST_NAME, SALARY, MOD(SALARY,5000)
FROM EMPLOYEES
WHERE JOB_ID='SA_REP' ;
```

Hasil:

Results Explain Describe Saved SQL History		
LAST_NAME	SALARY	MOD(SALARY,5000)
Tucker	10000	0
Bernstein	9500	4500
Hall	9000	4000
Olsen	8000	3000
Cambrault	7500	2500
Tuvault	7000	2000
King	10000	0
Sully	9500	4500
McEwen	9000	4000
Smith	8000	3000
More than 10 rows available. Increase rows selector to view more rows.		

10 rows returned in 0,00 seconds [CSV Export](#)

Penjelasan:

Untuk employee dengan last_name 'Tucker', salary=10000, Salary dibagi dengan 5000 menghasilkan hasil 2 tanpa sisa (lihat record 1, MOD(10000,5000) = 0)

Employee dengan last_name 'Bernstein' salarynya = 9500, dibagi dengan 5000 akan menghasilkan hasil 1 dengan sisa 4500 (MOD(9500,5000)=4500)

B.5. Bekerja Dengan Date**B.5.1. Format Date**

Basisdata Oracle menyimpan *date* dalam format internal numerik yang terdiri dari : century, year, month, day, hours, minutes dan seconds. Tipe data *date* ditampilkan dalam format DD-MMM-RR.

Contoh 8

- Contoh menampilkan last_name dan hire_date dari employees yang hire_date < 01-FEB-88. Perhatikan tampilan hasil di kolom hire_date.

```
SELECT last_name, hire_date FROM employees WHERE
hire_date < '01-FEB-88';
```

Hasil:

LAST_NAME	HIRE_DATE
King	17-JUN-87
Whalen	17-SEP-87

Contoh 9

- Untuk menampilkan tanggal hari ini dapat menggunakan SYSDATE.

```
SELECT SYSDATE from dual;
```

Hasil:

SYSDATE
16-SEP-10

B.5.2. Aritmatika dengan Date

Karena basisdata Oracle menyimpan fungsi *date* dalam bentuk numerik, maka dapat dilakukan operasi aritmatika terhadap fungsi *date* seperti melakukan penambahan maupun pengurangan. Tabel 4.5 merupakan bentuk dan hasil operasi aritmatika terhadap fungsi *date*.

Tabel 4.5. Operasi aritmetika dengan Date

Operasi	Tipe Data Hasil	Keterangan
date + number	date	Menambahkan sejumlah hari ke dalam tanggal tertentu
date – number	date	Mengurangi sejumlah hari dari tanggal tertentu
date – date	jumlah hari (numerik)	Mengurangi satu tanggal dengan tanggal yang lain
date + number / 24	date	Menambah sejumlah jam ke dalam tanggal

Contoh 10

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

**Hasil:**

| LAST_NAME | WEEK       |
|-----------|------------|
| King      | 1213.2046  |
| Kochhar   | 1095.06174 |
| De Haan   | 922.204597 |

**Penjelasan:**

Menampilkan last\_name dan telah berapa minggu employee telah bekerja untuk seluruh employee yang bekerja di departement 90.

Catatan : Hasil yang diperoleh akan berbeda tergantung dari SYSDATE yang digunakan.

**B.5.3. Berbagai Fungsi Date**

Semua fungsi DATE dalam ORACLE akan mengembalikan nilai bertipe DATE kecuali MONTHS\_BETWEEN yang akan mengembalikan tipe data numerik. Tabel 6 merupakan tabel untuk beberapa fungsi date.

**Tabel 4.6. Tabel Fungsi Date, Kegunaan, Contoh dan Hasil Contoh.**

**Catatan : Asumsi Sysdate : 23-AUG-17**

| FUNGSI                       | TUJUAN                                                                                                                                                           | CONTOH FUNGSI                           | HASIL FUNGSI |
|------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------|-----------------------------------------|--------------|
| MONTHS_BETWEEN(date1, date2) | Mengembalikan jumlah bulan antara <i>date1</i> dengan <i>date2</i> .                                                                                             | MONTHS_BETWEEN('17-AUG-45','17-AUG-17') | 336          |
| ADD_MONTHS (date,n)          | Menambahkan sejumlah n bulan ke dalam tanggal. Nilai n boleh negatif                                                                                             | ADD_MONTHS('02-JUL-17',6)               | 02-JAN-18    |
| NEXT_DAY(date,'char')        | Mendapatkan tanggal dari hari setelah tanggal yang ditetapkan. Hari yang dimasukan dalam 'char' dapat berupa nilai numerik dari hari maupun nilai karakter hari. | NEXT_DAY('04-SEP-17','FRIDAY'))         | 08-SEP-17    |
| LAST_DAY(date)               | Mengembalikan tanggal akhir bulan dalam date.                                                                                                                    | LAST_DAY('01-FEB-17')                   | 28-FEB-17    |
| ROUND(date,'MONTH')          | Pembulatan date ke Bulan Terdekat                                                                                                                                | Round(SYSDATE,'MONTH')                  | 01-SEP-17    |
| ROUND(date,'YEAR')           | Pembulatan date ke tahun terdekat                                                                                                                                | Round(sysdate,'YEAR')                   | 01-JAN-18    |
| TRUNC(date,'MONTH')          | Pemotongan date sampai bulan                                                                                                                                     | TRUNC(sysdate,'MONTH')                  | 01-AUG-17    |
| TRUNC(date,'YEAR')           | Pemotongan date sampai tahun                                                                                                                                     | TRUNC(sysdate,'YEAR')                   | 01-JAN-17    |

Berikut contoh-contoh penggunaan fungsi DATE dalam perintah SELECT

**Contoh 11**

- Ingin menampilkan employee id, hire date, tanggal evaluasi (TENURE) karyawan setelah bekerja selama 6 bulan, hari Jumat pertama setelah tanggal hire date, tanggal terakhir dari bulan hire date untuk seluruh employee yang telah bekerja lebih dari 200 bulan.

```
SELECT employee_id, hire_date,
 months_between(SYSDATE,hire_date) TENURE,
 add_months(hire_date,6) REVIEW,
 next_day(hire_date,'FRIDAY'),
 last_day(hire_date)
from employees
where months_between (sysdate,hire_date) > 200;
```

**Hasil :**

| EMPLOYEE_ID | HIRE_DATE | TENURE     | REVIEW    | NEXT_DAY( | LAST_DAY( |
|-------------|-----------|------------|-----------|-----------|-----------|
| 100         | 17-JUN-87 | 278.981863 | 17-DEC-87 | 19-JUN-87 | 30-JUN-87 |
| 101         | 21-SEP-89 | 251.85283  | 21-MAR-90 | 22-SEP-89 | 30-SEP-89 |
| 102         | 13-JAN-93 | 212.110895 | 13-JUL-93 | 15-JAN-93 | 31-JAN-93 |
| 103         | 03-JAN-90 | 248.433476 | 03-JUL-90 | 05-JAN-90 | 31-JAN-90 |
| 104         | 21-MAY-91 | 231.85283  | 21-NOV-91 | 24-MAY-91 | 31-MAY-91 |
| 200         | 17-SEP-87 | 275.981863 | 17-MAR-88 | 18-SEP-87 | 30-SEP-87 |

**Contoh 12**

- Membandingkan hire date dari seluruh karyawan yang bekerja mulai di tahun 1997. Yang ditampilkan adalah employee id, hire\_date, awal bulan mulai kerja dengan fungsi ROUND dan TRUNC.

```
SELECT employee_id,hire_date,
 ROUND(hire_date,'MONTH'),TRUNC(hire_date,'MONTH')
FROM employees
WHERE hire_date like '%97';
```

**Hasil :**

| EMPLOYEE_ID | HIRE_DATE | ROUND(HIR | TRUNC(HIR |
|-------------|-----------|-----------|-----------|
| 105         | 25-JUN-97 | 01-JUL-97 | 01-JUN-97 |
| 110         | 28-SEP-97 | 01-OCT-97 | 01-SEP-97 |
| 111         | 30-SEP-97 | 01-OCT-97 | 01-SEP-97 |
| 116         | 24-DEC-97 | 01-JAN-98 | 01-DEC-97 |
| 117         | 24-JUL-97 | 01-AUG-97 | 01-JUL-97 |
| 121         | 10-APR-97 | 01-APR-97 | 01-APR-97 |
| 123         | 10-OCT-97 | 01-OCT-97 | 01-OCT-97 |
| 125         | 16-JUL-97 | 01-AUG-97 | 01-JUL-97 |
| 129         | 20-AUG-97 | 01-SEP-97 | 01-AUG-97 |
| 130         | 30-OCT-97 | 01-NOV-97 | 01-OCT-97 |
| 131         | 16-FEB-97 | 01-MAR-97 | 01-FEB-97 |

**D. TUGAS**

**Catatan :** Untuk setiap nomor simpanlah sebagai sebuah file dengan format lab4\_xx (dimana xx adalah nomor tugas).

- Buatlah kueri untuk menampilkan last name employee dengan cara huruf pertama huruf besar dan huruf yang lain huruf kecil, beserta panjang dari last name tersebut untuk semua

employee yang last namanya diawali dengan huruf J, A, atau M. Urutkan hasilnya berdasarkan last name.

| EMPLOYEE_ID | LAST_NAME | LENGTH(LAST_NAME) |
|-------------|-----------|-------------------|
| 174         | Abel      | 4                 |
| 166         | Ande      | 4                 |
| 130         | Atkinson  | 8                 |
| 105         | Austin    | 6                 |
| 179         | Johnson   | 7                 |
| 195         | Jones     | 5                 |
| 133         | Mallin    | 6                 |
| 128         | Markle    | 6                 |
| 131         | Marlow    | 6                 |
| 164         | Marvins   | 7                 |

More than 10 rows available. Increase rows selector to view more rows.

10 rows returned in 0,00 seconds [CSV Export](#)

2. Buatlah kueri untuk menampilkan employee id, first name, first name dari employee dengan cara huruf pertama huruf kecil dan huruf yang lain huruf besar beri nama kolom ini dengan first\_name2 beserta dengan posisi huruf a dalam first name beri nama kolom ini dengan 'Contains 'a'?' . Yang ditampilkan hanya untuk employee yang memiliki first name yang diakhiri dengan huruf n urutkan berdasarkan first name.

|     |          |          |   |
|-----|----------|----------|---|
| 158 | Allan    | aLLAN    | 4 |
| 114 | Den      | dEN      | 0 |
| 174 | Ellen    | eLLEN    | 0 |
| 169 | Harrison | hARRISON | 2 |
| 204 | Hermann  | hERMANN  | 5 |
| 133 | Jason    | jASON    | 2 |
| 181 | Jean     | jEAN     | 3 |
| 110 | John     | jOHN     | 0 |
| 145 | John     | jOHN     | 0 |
| 139 | John     | jOHN     | 0 |
| 176 | Jonathon | jONATHON | 4 |
| 119 | Karen    | kAREN    | 2 |
| 146 | Karen    | kAREN    | 2 |
| 124 | Kevin    | kEVIN    | 0 |
| 197 | Kevin    | kEVIN    | 0 |

More than 15 rows available. Increase rows selector to view more rows.

15 rows returned in 0,00 seconds [CSV Export](#)

3. Buatlah kueri untuk menampilkan last name dan salary untuk semua employee yang mempunyai job sebagai SA\_REP. Format salary dengan panjang karakter 10 dengan left-padded (rata kanan) dan isi karakter di kiri dengan \$. Ditampilkan berdasarkan last name.

| Results Explain Describe Saved SQL History                             |           |                 |
|------------------------------------------------------------------------|-----------|-----------------|
| EMPLOYEE_ID                                                            | LAST_NAME | SALARY          |
| 174                                                                    | Abel      | \$\$\$\$\$11000 |
| 166                                                                    | Ande      | \$\$\$\$\$6400  |
| 130                                                                    | Atkinson  | \$\$\$\$\$2800  |
| 105                                                                    | Austin    | \$\$\$\$\$4800  |
| 204                                                                    | Baer      | \$\$\$\$\$10000 |
| 116                                                                    | Baida     | \$\$\$\$\$2900  |
| 167                                                                    | Banda     | \$\$\$\$\$6200  |
| 172                                                                    | Bates     | \$\$\$\$\$7300  |
| 192                                                                    | Bell      | \$\$\$\$\$4000  |
| 151                                                                    | Bernstein | \$\$\$\$\$9500  |
| More than 10 rows available. Increase rows selector to view more rows. |           |                 |
| 10 rows returned in 0,00 seconds <a href="#">CSV Export</a>            |           |                 |

4. Untuk setiap employee, tampilkan employee id, last name, salary dan salary yang telah dinaikan sebanyak 17%. Beri nama kolom terakhir dengan NEW SALARY. Bulatkan hasil NEW SALARY, 2 digit setelah titik desimal. Diurutkan berdasarkan NEW SALARY tertinggi.

| EMPLOYEE_ID | LAST_NAME | SALARY | NEW SALARY |
|-------------|-----------|--------|------------|
| 100         | King      | 24000  | 28080      |
| 101         | Kochhar   | 17000  | 19890      |
| 102         | De Haan   | 17000  | 19890      |
| 145         | Russell   | 14000  | 16380      |
| 146         | Partners  | 13500  | 15795      |
| 201         | Hartstein | 13000  | 15210      |
| 108         | Greenberg | 12000  | 14040      |
| 205         | Higgins   | 12000  | 14040      |
| 147         | Errazuriz | 12000  | 14040      |
| 168         | Ozer      | 11500  | 13455      |
| 174         | Abel      | 11000  | 12870      |

5. Modifikasi kueri tugas 4 dengan menambah kolom yang berisi salary baru dikurangi dengan salary lama. Beri nama kolom baru ini dengan nama INCREASE. Urutkan dari INCREASE yang tertinggi.

| EMPLOYEE_ID | LAST_NAME | SALARY | NEW SALARY | INCREASE |
|-------------|-----------|--------|------------|----------|
| 100         | King      | 24000  | 28080      | 4080     |
| 101         | Kochhar   | 17000  | 19890      | 2890     |
| 102         | De Haan   | 17000  | 19890      | 2890     |
| 145         | Russell   | 14000  | 16380      | 2380     |
| 146         | Partners  | 13500  | 15795      | 2295     |
| 201         | Hartstein | 13000  | 15210      | 2210     |
| 108         | Greenberg | 12000  | 14040      | 2040     |
| 205         | Higgins   | 12000  | 14040      | 2040     |
| 147         | Errazuriz | 12000  | 14040      | 2040     |
| 168         | Ozer      | 11500  | 13455      | 1955     |
| 174         | Abel      | 11000  | 12870      | 1870     |

6. Untuk setiap employee, tampilkan last\_name, hitung berapa bulan employee tersebut telah bekerja yang dihitung dari hire date sampai dengan sekarang. Beri nama kolom dengan

MONTHS\_WORKED. Tampilkan berdasarkan MONTHS\_WORKED terendah. Bulatkan hasil perhitungan MONTHS\_WORKED nya (tidak ada nilai desimal).

Catatan: hasil akan berbeda-beda tergantung sysdate.

| LAST_NAME  | MONTHS WORKED |
|------------|---------------|
| Smith      | 5             |
| Kumar      | 125           |
| Banda      | 125           |
| Ande       | 126           |
| Markle     | 126           |
| Lee        | 127           |
| Grant      | 128           |
| Geoni      | 128           |
| Marvins    | 128           |
| Zlotkey    | 128           |
| Philtanker | 128           |

7. Ingin dibuat laporan dari seluruh employee beserta dengan masa kerjanya. Buatlah kueri untuk menampilkan last name beserta dengan jumlah masa kerja dalam tahun dan jumlah bulan yang telah dilaluinya. Tampilkan berdasarkan lama kerja, yang memiliki masa kerja paling lama akan ditampilkan terlebih dahulu.

Results Explain Describe Saved SQL History

| LAST_NAME                                                              | TAHUN | BULAN |
|------------------------------------------------------------------------|-------|-------|
| King                                                                   | 23    | 3     |
| Whalen                                                                 | 23    | 0     |
| Kochhar                                                                | 21    | 0     |
| Hunold                                                                 | 20    | 8     |
| Ernst                                                                  | 19    | 4     |
| De Haan                                                                | 17    | 8     |
| Mavris                                                                 | 16    | 3     |
| Baer                                                                   | 16    | 3     |
| Higgins                                                                | 16    | 3     |
| Gietz                                                                  | 16    | 3     |
| More than 10 rows available. Increase rows selector to view more rows. |       |       |

10 rows returned in 0,02 seconds

[CSV Export](#)

## E. DAFTAR PUSTAKA

1. *Oracle Database 10g : SQL Fundamental I*, Oracle Inc. 2004

☺ Kerjakan segala pekerjaan dengan hati gembira maka hasil pekerjaan akan menggembarakan hati ☺