



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

**MODUL 2
BAHASA QUERY**

Membatasi Data dengan Klausa WHERE

A. TUJUAN

- Mahasiswa dapat membatasi baris yang dikembalikan oleh sebuah kueri (*query*) dengan menggunakan klausa WHERE.

B. LANDASAN TEORI & LANGKAH PRAKTIKUM

Ketika mengambil data dari basisdata, kadang diinginkan untuk membatasi baris data yang ditampilkan atau mengurutkan baris yang ditampilkan. Modul ini menjelaskan perintah SQL untuk melakukan hal tersebut.

B.1. Membatasi Baris Menggunakan Sebuah Seleksi

Gambar 2.1 berikut ini menggambarkan contoh untuk menampilkan semua employee dari department 90. Dengan demikian, hanya baris-baris data dengan kolom DEPARTMENT_ID yang nilainya 90 yang akan ditampilkan. Untuk membatasi baris yang dikembalikan dari kueri, digunakan klausa WHERE dalam SQL.

EMPLOYEES

EMPLOYEE_ID	LAST_NAME	JOB_ID	DEPARTMENT_ID
100	King	AD_PRES	90
101	Kochhar	AD_VP	90
102	De Haan	AD_VP	90
103	Hunold	IT_PROG	60
104	Ernst	IT_PROG	60
107	Lorentz	IT_PROG	60
124	Mourgos	ST_MAN	50

...
20 rows selected.

“retrieve all employees in department 90”

EMPLOYEE_ID	LAST_NAME	JOB_ID	DEPARTMENT_ID
100	King	AD_PRES	90
101	Kochhar	AD_VP	90
102	De Haan	AD_VP	90

Gambar 2.1. Membatasi Baris dengan Seleksi
Sumber: Oracle 2004

B.2. Sintaks umum dari klausa WHERE

```
SELECT    * | {[DISTINCT] column/expression [alias],...}
FROM      table
[WHERE    condition(s)];
```

Keterangan:

- **WHERE** membatasi kueri yang terhadap baris yang sesuai dengan kondisi
- *condition* terdiri atas nama kolom, ekspresi, konstanta, dan operator perbandingan

Klausula WHERE berisi suatu kondisi yang harus dipenuhi. Klausula WHERE langsung mengikuti klausula FROM dalam perintah SELECT. Jika kondisi benar, maka baris yang memenuhi kondisi akan dikembalikan.

Klausula WHERE dapat membandingkan nilai-nilai dalam kolom, nilai literal, ekspresi matematika, atau fungsi. Klausula ini terdiri atas:

- nama kolom
- kondisi perbandingan
- nama kolom, konstanta, atau daftar nilai.

Contoh 1

- Cobalah contoh klausula WHERE berikut ini

```
SELECT employee_id, last_name, job_id, department_id
FROM employees
WHERE department_id = 90;
```

Hasil :

EMPLOYEE ID	LAST NAME	JOB ID	DEPARTMENT ID
100	King	AD_PRES	90
101	Kochhar	AD_VP	90
102	De Haan	AD_VP	90

Dalam contoh di atas, perintah SELECT akan mengembalikan employee ID, last name, job ID, dan department ID dari semua employee yang memiliki department_id 90.

B.3. Karakter *String* dan *Date*

Karakter *String* dan *Date* dalam klausula WHERE harus ditandai dengan tanda petik tunggal (' '). Konstanta bilangan tidak memerlukan tanda petik tunggal ini. **Semua karakter yang dicari dalam Oracle SQL bersifat *case sensitive*.**

Contoh 2

- Cobalah jalankan perintah berikut ini dan amati data apa yang ditampilkan.

```
SELECT last_name, job_id, department_id
FROM employees
WHERE last_name = 'Whalen';
```

Hasil :

Results:		
LAST_NAME	JOB_ID	DEPARTMENT_ID
1 Whalen	AD_ASST	10

Contoh 3

- Modifikasilah contoh 2 menjadi perintah seperti ini dan amati apa yang terjadi. Bandingkan hasilnya dengan contoh 2.

```
SELECT last_name, job_id, department_id
FROM employees
WHERE last_name = 'WHALEN';
```

Hasil :

Results:		
LAST_NAME	JOB_ID	DEPARTMENT_ID

Basisdata Oracle menyimpan data tanggal dalam format numerik internal, yang merepresentasikan abad, tahun, bulan, tanggal, jam, menit, dan detik. Tanggal default ditampilkan dalam bentuk tanggal-bulan-tahun (DD-MON-RR). Perubahan format tanggal akan dibahas secara khusus dalam modul praktikum selanjutnya.

B.4. Kondisi Perbandingan

Kondisi perbandingan dipergunakan untuk membandingkan suatu ekspresi dengan nilai atau ekspresi yang lain. Kondisi perbandingan dipergunakan dalam klausa WHERE dengan format sebagai berikut:

... WHERE *expr operator value*

Keterangan:

- *expr* adalah ekspresi yang akan dibandingkan
- *operator* adalah operator pembanding
- *value* adalah nilai yang akan dibandingkan dengan ekspresi

Operator pembanding yang dikenal dalam Oracle SQL adalah seperti dalam tabel 2.1 berikut ini:

Tabel 2.1 Operator Pembanding

OPERATOR	ARTI
=	Sama dengan
>	Lebih besar dari
>=	Lebih besar dari atau sama dengan
<	Lebih kecil dari
<=	Lebih kecil dari atau sama dengan
<>	Tidak sama dengan
BETWEEN AND	Antara dua nilai (inklusif)
IN (himpunan)	Sesuai dengan salah satu nilai dalam daftar anggota himpunan
LIKE	Sesuai dengan suatu pola karakter
IS NULL	Bernilai null

Catatan:

- Sebuah nama alias tidak bisa digunakan dalam klausa WHERE.
- Simbol **!=** dan **^=** dapat juga digunakan untuk menyatakan kondisi TIDAK SAMA.

Contoh 4

- Cobalah contoh klausa WHERE dengan kondisi perbandingan berikut

```
SELECT last_name, salary
FROM employees
WHERE salary <= 3000;
```

Hasil:



	LAST_NAME	SALARY
1	Baida	2900
2	Tobias	2800
3	Himuro	2600
4	Colmenares	2500
5	Mikkilineni	2700
6	Landry	2400
7	Markle	2200
8	Atkinson	2800
9	Marlow	2500
10	Olson	2100
11	Rogers	2900
12	Gee	2400
13	Philtanker	2200
14	Seo	2700
15	Patel	2500

Dalam contoh di atas, perintah SELECT akan menampilkan nama dan salary dari tabel EMPLOYEES, di mana salary dari employee yang ditampilkan adalah yang nilainya kurang atau sama dengan 3000. Nilai konstanta 3000 dibandingkan dengan nilai salary dari kolom SALARY dalam tabel EMPLOYEES.

Contoh 5

- Modifikasilah klausa WHERE dalam contoh 4 di atas dengan klausa-klausa WHERE berikut (satu perintah SELECT untuk satu klausa WHERE) dan amati hasilnya.

```
... WHERE hire_date = '07-JUN-94'
... WHERE salary >= 6000
... WHERE last_name = 'Smith'
```

B.5. Menggunakan Kondisi BETWEEN

Kondisi BETWEEN dipergunakan untuk menampilkan baris berdasarkan jangkauan nilai tertentu. Jangkauan dispesifikasikan dengan batas bawah dan batas atas.

Contoh 6

- Cobalah contoh kondisi BETWEEN berikut ini:

```
SELECT last_name, salary
FROM employees
WHERE salary BETWEEN 2500 AND 3500;
```

Batas bawah

Batas atas

Hasil:



	LAST_NAME	SALARY
1	Khoo	3100
2	Baida	2900
3	Tobias	2800
4	Himuro	2600
5	Colmenares	2500
6	Nayer	3200
7	Mikkilineni	2700
8	Bissot	3300
9	Atkinson	2800
10	Marlow	2500
11	Mallin	3300
12	Rogers	2900
13	Stiles	3200
14	Seo	2700
15	Patel	2500

Perintah SELECT di atas akan menampilkan employee yang memiliki salary di antara 2500 dan 3500. Nilai yang dispesifikasikan dalam kondisi BETWEEN harus dimulai dari batas bawah lebih dahulu.

B.6. Menggunakan Kondisi IN

Kondisi IN dipergunakan untuk mengecek apakah suatu nilai tertentu termasuk dalam sekumpulan nilai yang dispesifikasikan. Kondisi IN juga dikenal sebagai kondisi keanggotaan (*membership*). Kondisi IN dapat dipergunakan untuk beberapa tipe data. Jika data yang digunakan dalam IN bertipe karakter atau date, maka harus diletakkan diantara tanda petik tunggal.

Contoh 7

- Cobalah contoh kondisi IN berikut ini:

```
SELECT employee_id, last_name, salary, manager_id
FROM employees
WHERE manager_id IN (100,101,201);
```

Hasil:

EMPLOYEE_ID	LAST_NAME	SALARY	MANAGER_ID
1	101 Kochhar	17000	100
2	102 De Haan	17000	100
3	114 Raphaely	11000	100
4	120 Weiss	8000	100
5	121 Fripp	8200	100
6	122 Kaufing	7900	100
7	123 Vollman	6500	100
8	124 Mourgos	5800	100
9	145 Russell	14000	100
10	146 Partners	13500	100
11	147 Errazuriz	12000	100
12	148 Cambrault	11000	100
13	149 Zlotkey	10500	100
14	201 Hartstein	13000	100
15	108 Greenberg	12000	101

Perintah SELECT di atas akan menampilkan employee ID, nama, salary, manager ID dari employee yang memiliki manager ID 100, 101, atau 201.

Contoh 8

- Cobalah contoh kondisi IN dengan data bertipe karakter berikut ini dan amati hasilnya.

```
SELECT employee_id, last_name, manager_id, department_id
FROM employees
WHERE last_name IN ('Hartstein', 'Vargas');
```

Hasil:

EMPLOYEE_ID	LAST_NAME	MANAGER_ID	DEPARTMENT_ID
1	201 Hartstein	100	20
2	144 Vargas	124	50

B.7. Menggunakan Kondisi LIKE

Tidak setiap dalam setiap pencarian diketahui nilai pasti yang dicari. Dapat saja dilakukan pencarian terhadap suatu pola karakter yang sesuai dengan menggunakan kondisi LIKE. Operasi pencocokan pola karakter yang sesuai ini sering disebut sebagai pencarian **wildcard**. Terdapat dua simbol yang dipergunakan untuk membentuk pencarian string yaitu:

- % untuk merepresentasikan rangkaian **beberapa karakter** maupun karakter kosong
- _ untuk merepresentasikan **satu karakter tunggal**

Contoh 9

- Cobalah contoh kondisi LIKE berikut ini dan amati hasilnya:

```
SELECT first_name
FROM employees
WHERE first_name LIKE 'S%';
```

Hasil:

Results:

	FIRST_NAME
1	Sundar
2	Shelli
3	Sarah
4	Shelley
5	Steven
6	Sundita
7	Steven
8	Susan
9	Samuel
10	Sarath
11	Stephen
12	Sigal
13	Shanta

SQL History

Contoh 10

- Kondisi LIKE dapat pula dipergunakan sebagai *shortcut* untuk beberapa kondisi BETWEEN seperti dalam contoh berikut ini yang akan menampilkan nama dan hire date dari employee yang mulai bekerja antara Januari 1995 dan Desember 1995.

```
SELECT last_name, hire_date
FROM employees
WHERE hire_date LIKE '%95';
```

Hasil:

Results:

	LAST_NAME	HIRE_DATE
1	Khoo	18-MAY-95
2	Kaufling	01-MAY-95
3	Ladwig	14-JUL-95
4	Rajs	17-OCT-95

Kombinasi Karakter Wildcard

Simbol % dan _ dalam kondisi LIKE dapat dikombinasikan dengan karakter literal.

Contoh 11

- Cobalah contoh berikut ini yang akan menampilkan nama employee yang huruf kedua dalam namanya adalah o.

```
SELECT last_name
FROM employees
WHERE last_name LIKE '_o%';
```

Hasil:

Results:

	LAST_NAME
1	Colmenares
2	Doran
3	Fox
4	Johnson
5	Jones
6	Kochhar
7	Lorentz
8	Mourgos
9	Popp
10	Rogers
11	Tobias
12	Vollman

Opsi ESCAPE

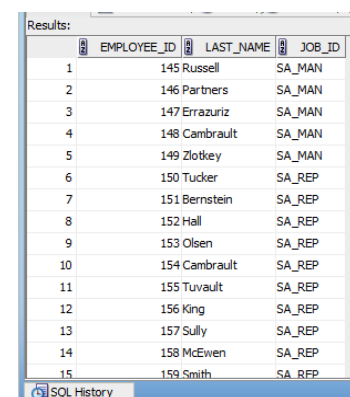
Opsi ESCAPE dapat dipergunakan untuk mendapatkan suatu nilai yang memuat karakter % dan _. Misalnya jika diinginkan untuk menampilkan string yang mengandung 'SA_ '.

Contoh 12

- Cobalah contoh berikut ini yang akan menampilkan employee ID, nama employee, dan job ID dari employee yang job_ID nya memuat 'SA_ '. Perhatikan bahwa karakter escape '\' dituliskan sebelum '_' dan opsi ESCAPE dituliskan setelahnya untuk membuat server Oracle dapat mengartikan tanda '_' sebagaimana adanya.

```
SELECT employee_id, last_name, job_id
FROM   employees
WHERE  job_id LIKE 'SA\_%' ESCAPE '\';
```

Hasil:



	EMPLOYEE_ID	LAST_NAME	JOB_ID
1	145	Russell	SA_MAN
2	146	Partners	SA_MAN
3	147	Errazuriz	SA_MAN
4	148	Cambrault	SA_MAN
5	149	Zlotkey	SA_MAN
6	150	Tucker	SA_REP
7	151	Bernstein	SA_REP
8	152	Hall	SA_REP
9	153	Olsen	SA_REP
10	154	Cambrault	SA_REP
11	155	Tuvault	SA_REP
12	156	King	SA_REP
13	157	Sully	SA_REP
14	158	McEwen	SA_REP
15	159	Smith	SA_REP

B.8. Menggunakan Kondisi NULL

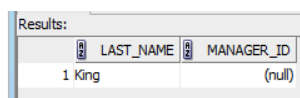
Kondisi NULL mencakup kondisi IS NULL dan kondisi IS NOT NULL. Kondisi IS NULL dipergunakan untuk mengetes null. Sebuah nilai null berarti nilai tersebut tidak tersedia (*unavailable, unassigned, unknown, atau inapplicable*). Jadi, null tidak bisa dites dengan menggunakan tanda '=' karena nilai null tidak sama dengan nilai apapun.

Contoh 13

- Cobalah contoh berikut ini yang akan menampilkan nama employee dan manager ID dari employee yang tidak memiliki manager.

```
SELECT last_name, manager_id
FROM   employees
WHERE  manager_id IS NULL;
```

Hasil:



	LAST_NAME	MANAGER_ID
1	King	(null)

Contoh 14

- Cobalah contoh berikut ini yang akan menampilkan nama employee, job ID, dan komisi dari seluruh employee yang tidak mendapatkan komisi.

```
SELECT last_name, job_id, commission_pct
FROM   employees
WHERE  commission_pct IS NULL;
```

Hasil:

Results:			
	LAST_NAME	JOB_ID	COMMISSION_PCT
1	King	AD_PRES	(null)
2	Kochhar	AD_VP	(null)
3	De Haan	AD_VP	(null)
4	Hunold	IT_PROG	(null)
5	Ernst	IT_PROG	(null)
6	Austin	IT_PROG	(null)
7	Pataballa	IT_PROG	(null)
8	Lorentz	IT_PROG	(null)
9	Greenberg	FI_MGR	(null)
10	Faviet	FI_ACCOUNT	(null)
11	Chen	FI_ACCOUNT	(null)
12	Sciarra	FI_ACCOUNT	(null)
13	Urman	FI_ACCOUNT	(null)
14	Popp	FI_ACCOUNT	(null)
15	Raphaely	PII_MAN	(null)

D. DAFTAR PUSTAKA

Oracle Database 19g : SQL Fundamental, Oracle Inc. 2023

😊 Baik itu tidak cukup jika lebih baik itu mungkin... 😊