|  |  |
| --- | --- |
|  | D:\Dokumen Mocher\desktop\logo UMB.jpg |
|  | **MODUL PERKULIAHAN** |
|  |  |
|  | **Basis Data** |
|  |  |
|  | SQL-Data Manipulation Language – Query Multi Tabel |
|  |  |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |  |
|  | **Fakultas** | **Program Studi** | **Tatap Muka** | **Kode MK** | **Disusun Oleh** |  |
|  | FASILKOM | TEKNIK INFORMATIKA | **05** |  | Baby Lolita Basyah, SKom.,MMSI |  |

|  |  |
| --- | --- |
| **Abstract** | **Kompetensi** |
|  |  |
| Pada materi ini akan dibahas perintah SQL SELECT dalam sub query dan join table , serta penggunaan perintah INSERT, UPDATE dan DELETE untuk memanipulasi data. | Mahasiswa diharapkan dapat menulis perintah SQL SELECT dalam sub queri dan join table dan mampu memanipulasi data dengan menggunakan perintah INSERT, UPDATE, dan DELETE. |

# Structure Query Language (SQL)

*Structure Query Language* (SQL) merupakan komponen bahasa *relational database system*. SQL merupakan bahasa baku (ANSI/SQL), *non procedural*, dan berorientasi himpunan (*set-oriented language*). SQL dapat digunakan baik secara interaktif atau ditempelkan (*embedded*) pada sebuah program aplikasi.

## DATA MANIPULATION LANGUAGE

Perintah yang termasuk DML meliputi :

**1. INSERT**

Fungsi : menambah baris (record) baru

Sintaks : **INSERT INTO tbname**

**(col1, ...) VALUES (value1, ...)**

Catatan :

Sintaks tersebut dapat digunakan jika jumlah kolom = jumlah nilai, tetapi jika dalam tabel semua kolom akan diisi dapat digunakan sintaks berikut ini :

Sintaks : **INSERT INTO tbname**

**VALUES (value1, value2, ...)**

Nilai-nilai diisikan sebanyak kolom yang terdapat di tabel tersebut.

**2. UPDATE**

Fungsi : merubah record

Sintaks : **UPDATE tbname SET field = ekspresi**

**WHERE kondisi**

**3. DELETE**

Fungsi : menghapus record

Sintaks : **DELETE FROM tbname**

**WHERE kondisi**

**4. SELECT**

Fungsi : menampilkan record

Sintaks : SELECT [DISTINCT] colname FROM tbname

[WHERE kondisi] [GROUP BY kondisi] [HAVING kondisi] [ORDER BY kondisi]

**Contoh Kasus DML :**

Tabel Mahasiswa, MataKuliah dan Nilai berisikan dengan data-data sebagai berikut :

TABEL Mahasiswa

|  |  |  |  |
| --- | --- | --- | --- |
| **NIMmMSn** | **NAMAMHS** | **ALAMAT** | **City** |
| 1029 | Nuri | Jakarta | London |
| 3128 | Budi | Depok | Paris |
| 4129 | Ningrum | Bogor | Paris |
| 5010S4 | Pipit | Bekasi | London |
| 2111S5 | Adam | Bogor | Athens |

TABEL MataKuliah

|  |  |  |
| --- | --- | --- |
| **KdMK** | NamaMK | **SKS** |
| KK021 | Basis Data | 3 |
| KD132 | SIM | 2 |
| KU122 | Pancasila | 2 |

TABEL Nilai

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **NIM** | **KdMK** | **SKS** | **MID** | **FINAL** |
| 1029 | KK021 | 3 | 60 | 75 |
| 3128 | KK021 | 3 | 50 | 40 |
| 1029 | KU122 | 2 | 70 | 85 |
| 4129 | KU122 | 2 | 90 | 80 |
| 2111 | KU122 | 2 | 75 | 75 |
| 5010 | KD132 | 2 | 80 |  |
| 1029 | KD132 | 2 | 50 | 65 |
| 2111 | KD132 | 2 | 60 | 80 |

**Menambah record (INSERT)**

* Memasukkan data mata kuliah Perancangan Sistem dengan kode KK222 dan besarnya 4 sks

INSERT INTO MataKuliah

VALUES (‘KK222’, ‘Perancangan Sistem’, 4)

* Memasukkan data ke dalam table Nilai berupa NIM, KdMK, MID. NIM = 3122, KdMK = KK222 , MID = 40, dan FINAL = 60

INSERT INTO Nilai

VALUES (‘KK3122’,’KK222’, 40, 60)

**Merubah record (UPDATE)**

* Merubah alamat mahasiswa menjadi ‘DEPOK’ untuk mahasiswa yang memiliki NIM 5010

UPDATE Mahasiswa SET ALAMATMHS = ‘DEPOK’

WHERE NIM = ‘5010’

* Merubah nilai mahasiswa dimana nilai MIDnya ditambah 10 untuk KdMK ‘KK021

’ UPDATE Nilai SET MID = MID + 10

WHERE KdMK = ‘KK021’

**Menghapus record (DELETE)**

* Menghapus data nilai mata kuliah ‘KK021’ bagi mahasiswa yang mempunyai NIM 1029

DELETE FROM Nilai

WHERE NIM= ‘1029’ AND NoMK = ‘KK021’

## Menampilkan record (SELECT lebih dari satu tabel / JOIN)

Tabel S, P dan SP isikan dengan data-data sebagai berikut :

TABEL Supplier

|  |  |  |  |
| --- | --- | --- | --- |
| **Sn** | **Sname** | **Status** | **City** |
| S1 | Smith | 20 | London |
| S2 | Jones | 10 | Paris |
| S3 | Blake | 30 | Paris |
| S4 | Clark | 20 | London |
| S5 | Adams | 30 | Athens |

TABEL Part

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Pn** | **Pname** | **Warna** | **Weight** | **City** |
| P1 | Nut | Red | 12 | London |
| P2 | Bolt | Green | 17 | Paris |
| P3 | Screw | Blue | 17 | Rome |
| P4 | Screw | Red | 14 | London |
| P5 | Cam | Blue | 12 | Paris |
| P6 | Cog | Red | 19 | London |

TABEL SP

|  |  |  |
| --- | --- | --- |
| **Sn** | **Pn** | **qty** |
| S1 | P1 | 300 |
| S1 | P2 | 200 |
| S1 | P | 40 |
| S1 | P4 | 200 |
| S1 | P5 | 100 |
| S1 | P6 | 100 |
| S2 | P1 | 300 |
| S2 | P2 | 400 |
| S3 | P2 | 200 |
| S4 | P2 | 200 |
| S4 | P4 | 300 |
| S4 | P5 | 400 |

1. Menampilkan semua supplier dan part yang keduanya bertempat tinggal pada kota yang sama

SELECT Sn, Sname,Status, City , Pn, Pname, Warna, Weight

FROM Supplier, Part

WHERE S.City = P.City

2. Menampilkan nama supplier yang memasok barang dengan nomor part P2

SELECT Sname FROM S, SP

WHERE S.Sn = SP.Sn AND SP.Pn = 'P2'

3. Menampilkan nama supplier yang memasok part berwarna merah

SELECT Sname FROM S,SP, P WHERE S.Sn = SP.Sn

AND SP.Pn = P.Pn

AND P.COLOR = 'RED'

## Menampilkan record (SELECT lebih dari satu tabel / SELECT Bertingkat)

1. Menampilkan nama supplier yang memasok barang dengan nomor part P2

SELECT Sname FROM S WHERE Sn IN (SELECT Sn FROM SP WHERE Pn = 'P2')

Atau

SELECT Sname FROM S WHERE Sn = ANY (SELECT Sn FROM SP WHERE Pn = 'P2')

2. Menampilkan nama supplier yang memasok part berwarna merah

SELECT Sname FROM S WHERE Sn IN (SELECT Sn FROM SP WHERE Pn IN

(SELECT Pn FROM P WHERE Warna = 'Red'))

3. Menampilkan no.supplier dengan nilai status lebih kecil daripada nilai maksimum status yang ada pada tabel S

SELECT Sn FROM S WHERE Status < (SELECT MAX(Status) FROM S)

4. Menampilkan nama supplier yang tidak memasok barang dengan nomor part P2

SELECT Sname FROM S WHERE Sn NOT IN (SELECT Sn FROM SP WHERE Pn = 'P2')

5. Menampilkan semua nomor supplier yang sama lokasinya dengan S1

SELECT Sn FROM S WHERE CITY =

(SELECT CITY FROM S WHERE Sn = 'S1')

## Penggunaan Clausa UNION, INTERSECT dan EXCEPT dalam query

Dalam SQL kita dapat menggunakan operasi himpunan UNION (GABUNGAN) , INTERSECTION (IRISAN) dan DIFFERENCE (SELISIH) untuk mengkombinasikan hasil dari dua atau lebih query ke dalam satu table hasil.

* UNION dari dua table, A dan B, adalah sebuah table yang berisi gabungan seluruh baris yang ada baik di table A ataupun di table B.
* INTERSECTION dari dua table, A dan B, adalah sebuah table yang berisi baris yang ada di kedua table A dan B.
* DIFFERENCE dari dua table, adalah sebuah table yang berisi seluruh baris yang ada di table A tapi tidak ada di table B.

Tabel Branch dan Property adalah sebagai berikut :

Branch

|  |  |  |  |
| --- | --- | --- | --- |
| branchNo | street | city | postcode |
| B005 | 22 Deer Rd | London | SW1 |
| B007 | 16 Argyll St | Aberden | AB2 |
| B003 | 163 Main St | Glasgow | G11 |
| B004 | 32 Manse Rd | Bristol | BS9 |
| B002 | 56 Clover Dr | London | NW1 |

Property

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Proper  tyNo | Street | City | postcd | Type | Room | Rent | Owner  No | Staff  No | Branch  No |
| PA14 | 16Holhead | Aberden | AB7 | H | 6 | 650 | CO46 | SA9 | B007 |
| PL94 | 6 Argyll St | London | NW2 | F | 4 | 400 | CO87 | SL41 | B005 |
| PG4 | 6 Lawrenc | Glasgow | G119 | F | 3 | 350 | CO40 |  | B003 |
| PG36 | 2 Manor | Glasgow | G324 | F | 3 | 375 | CO93 | SG37 | B003 |
| PG21 | 18 Dale | Glasgow | G12 | H | 5 | 600 | CO87 | SG37 | B003 |
| PG16 | 5 Novar | Glasgow | G129 | F | 4 | 450 | CO93 | SG14 | B003 |

* Penggunaan UNION

Menampilkan seluruh kota yang ada baik pada table branch ataupun property

Perintahnya adalah :

(SELECT city FROM branch

WHERE city IS NOT NULL)

UNION

(SELECT city FROM property

WHERE city IS NOT NULL)

Atau

(SELECT \* FROM branch

WHERE city IS NOT NULL)

UNION CORRESPONDING BY city

(SELECT \* FROM property

WHERE city IS NOT NULL)

Hasilnya adalah :

|  |
| --- |
| City |
| London |
| Glasgow |
| Aberden |
| Bristol |

* Penggunaan INTERSECT

Menampilkan kota yang ada di kedua table baranch dan property

Perintahnya adalah :

(SELECT city FROM branch)

INTERSECT

(SELECT city FROM property)

Atau

(SELECT \* FROM branch)

INTERSECT CORRESPONDING BY city

(SELECT \* FROM property)

Hasilnya adalah :

|  |
| --- |
| City |
| Aberdeen |
| Glasgow |
| London |

* Penggunaan EXCEPT

Menampilkan seluruh kota yg ada pada table branch tapi tidak pada property

Perintahnya adalah :

(SELECT city FROM branch)

EXCEPT

(SELECT city FROM property

Atau

(SELECT \* FROM branch)

EXCEPT CORRESPONDING BY city

(SELECT \* FROM property)

Hasilnya adalah :

|  |
| --- |
| City |
| Bristol |

# Daftar Pustaka

1. Connoly, Thomas; Begg, Carolyn; Strachan, Anne;  ***Database Systems : A Practical Approach to Design, Implementation and Management***, 3rd edition, Addison Wesley, 2001.

2. Korth, H.; ***Database System Concept***, 4th edition, Mc Graw Hill, New York, 1991

3. Linda Marlinda; ***Sistem Basis Data;*** edisi 1, Andi Offset, Yogyakarta, 2004