**LAPORAN RESMI**

**MODUL III**

**(STORED PROCEDURE)**

**SISTEM MANAJEMEN BASIS DATA**

**NAMA : IMAM ARISHANDI IRFANTO**

**N.R.P : 220441100034**

**DOSEN : FITRI DAMAYANTI, S.Kom,M.Kom**

**ASISTEN : M. IQBAL FIRMANSYAH**

**TGL PRAKTIKUM : 26 APRIL 2024**

**Disetujui, Mei 2024**

**Asisten**

**M. IQBAL FIRMANSYAH**

**21.04.411.00084**

**LABORATORIUM BISNIS INTELIJEN SISTEM**

**PRODI SISTEM INFORMASI**

**JURUSAN TEKNIK INFORMATIKA**

**FAKULTAS TEKNIK**

**UNIVERSITAS TRUNOJOYO MADURA**

# BAB I PENDAHULUAN

## Latar Belakang

Stored procedure merupakan salah satu fitur penting dalam sistem manajemen basis data (DBMS) yang memungkinkan pengguna untuk menyimpan kumpulan instruksi SQL yang telah didefinisikan di dalam database. Latar belakang penggunaan stored procedure sangatlah beragam dan berguna dalam konteks pengembangan aplikasi dan manajemen data.

Selain efisiensi, stored procedure juga menyediakan tingkat abstraksi yang lebih tinggi untuk pengguna dalam mengakses dan memanipulasi data. Hal ini memungkinkan pengembang untuk mengimplementasikan logika bisnis kompleks di dalam database, mengurangi kompleksitas kode aplikasi, dan mempercepat proses pengembangan.

Keuntungan lain dari penggunaan stored procedure adalah kemampuannya untuk mendukung transaksi kompleks. Dalam lingkungan transaksional, stored procedure dapat digunakan untuk mengelola proses transaksi yang melibatkan beberapa operasi SQL.

## Tujuan

1. Mampu memahami dan membuat procedure dalam basis data dan mampu menggunakan perintah-perintah dalam stored procedure serta menyelesaikan operasi-operasi data spesifik dengan memanfaatkan stored procedure.

# BAB II DASAR TEORI

Stored Procedure adalah sebuah prosedur layaknya subprogram (subrutin) di dalam bahasa pemrograman reguler yang tersimpan di dalam katalog basis data. Beberapa kelebihan yang ditawarkan stored procedure antara lain : mengingkatakan performa, mereduksi trafik jaringan, reusable, dan meningkatkan kontrol sekuriti. Di balik kelebihan tersebut, stored procedure juga memiliki kekurangan.

## Sintaks stored procedure:

<create procedure statement> ::=

CREATE PROCEDURE <procedure name> ( [ <parameter list>

] )

<routine body>

<parameter list> ::=

<parameter specification> [ , <parameter specification> ]...

<parameter specification> ::=

[ IN | OUT | INOUT ] <parameter> <data type>

<routine body> ::= <begin-end block>

<begin-end block> ::=

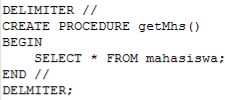
[ <label> : ] BEGIN <statement list> END [ <label> ]

<statement list> ::= { <body statement> ; }...

<statement in body> ::=

<declarative statement> | <procedural statement>

Pernyataan pembuatan stored procedure berikut :



Perintah DELIMITER digunakan untuk mengubah delimiter standar, misalnya di sini dari titik koma (;) menjadi slash ganda (//). Langkah ini umumnya dilakukan ketika isi stored procedure mengandung titik koma – yang merupakan delimiter standar di SQL. Pernyataan di antara BEGIN dan END merupakan badan (body) stored procedure. Perintah DELIMITER di akhir baris digunakan untuk mengembalikan delimiter ke karakter semula.

## Aktivasi/pemanggilan Stored Procedure

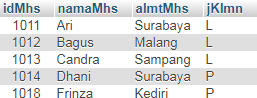
<call statement> ::=

CALL [ <database name> . ] <stored procedure name>

( [ <scalar expression> [ , <scalar expression> ]... ] )

Eksekusi Query tersebut dengan memanggil procedure getMahasiswa().





## Menghapus Stored Procedure

<drop procedure statement> ::= DROP PROCEDURE [ IF EXISTS ]

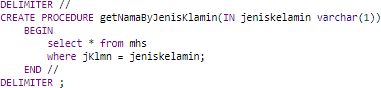
[ <database name> . ] <procedure name>

Dalam implementasinya, penggunaan stored procedure sering melibatkan parameter. Di MySQL, parameter stored procedure dibedakan menjadi tiga mode: IN, OUT, dan INOUT.

1. IN

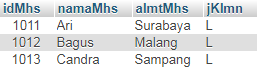
Parameter yang merupakan mode default ini mengindikasikan bahwa sebuah parameter dapat di-pass ke dalam stored procedure tetapi nilainya tidak dapat diubah (dari dalam stored procedure)

Sebagai contoh, kitab isa mendapatkan semua data matakuliah di semester tertentu.



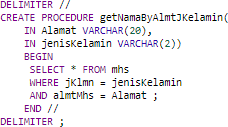
Untuk memanggil stored procedure yang memiliki parameter, maka kita harus menspesifikasikan argumenya. Misalkan kita ingin mendapatkan data mahasiswa dengan jenis kelamin laki-laki.

CALL getNamaByJenisKlamin(3);



Apabila pemanggilan stored procedure di atas mengabaikan argumen, DBMS akan merespon dengan pesan kesalahan.Bergantung kebutuhan, pendefinisian parameter pada stored procedure juga bisa lebih dari satu. Sebagai contoh, buat stored procedure dengan dua buah parameter

seperti berikut :



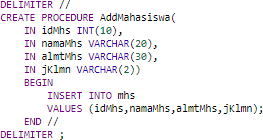
Pemanggilan stored procedure di atas tentunya akan memerlukan dua buah argumen.

CALL getNamaByAlmtJKelamin (‘surabaya’,’L’);



Penambahan Data

Pada operasi penambahan, data – data terkait diisikan melaui argumen. Selanjutnya, isi stored procedure akan memasukkan data ke dalam tabel. Berikut adalah contoh stored procedure untuk menambahkan data pada tabel mahasiswa.



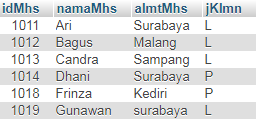
Lakukan eksekusi terhadap procedure tersebut

call AddMahasiswa('1019','Gunawan','surabaya', ‘L’);

Selanjutnya lakukan pengecekan data pada tabel mahasiswa.

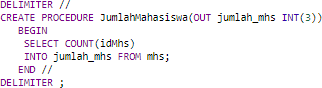
select \* from mhs; atau

select \* getMhs();



1. OUT

Mode ini mengindikasikan bahwa stored procedure dapat mengubah parameter dan mengirimkan kembali ke program pemanggil. Dalam konteks bahasa pemrograman, parameter OUT analog dengan passing-byreference. Dengan demikian, parameter ini nilainya bisa diubah oleh stored procedure.



Untuk mengeksekusi stored procedure dengan parameter OUT, dibutuhkan argumen yang spesifik.

call JumlahMahasiswa(@jumlah\_mhs);

Perhatikan, argumen harus menggunakan notasi @, yang mengindikasikan sebagai suatu parameter OUT.

Langkah selanjutnya, untuk mendapatkan nilai variabel, gunakan pernyataan SELECT

select @jumlah\_mhs;

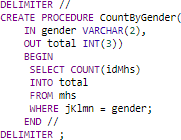


Parameter mode OUT juga bisa dikombinasikan dengan mode IN.

1. OUT

Mode ini pada dasarnya merupakan kombinasi dari mode IN dan OUT. kita bisa mengirimkan parameter kedalam stored procedure dan mendapatkan nilai kembalian yang baru dari stored procedure yang didefinisikan.

Sebagai contoh, definisikan stored procedure seperti berikut :



Lakukan eksekusi pada procedure tersebut untuk mencari jumlah mahasiswa yang berjenis kelamin perempuan .

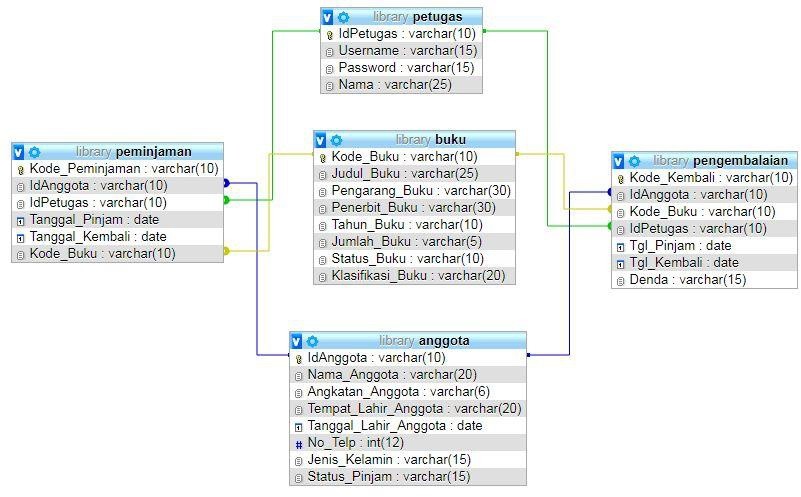




Stored procedure dapat mencerminkan beragam operasi data, misalnya seleksi, penambahan, pengubahan, penghapusan, dan juga operasi – oprasi DDL. Seperti halnya procedure di dalam bahasa pemrograman, stored procedure juga dapat melibatkan variabel, pernyataan kondisional, dan pengulangan.

# BAB IV IMPLEMENTASI

## Soal



1. Buatlah sebuah prosedur dengan menggunakan parameter IN default
2. Buatlah stored procedure untuk mengetahui data pada table buku berdasarkan pada salah satu field yaitu penerbit buku
3. Butalah stored procedure untuk mengetahui data pada table peminjaman berdasarkan dua field yaitu tanggal peminjaman dan kode buku
4. Buatlah stored procedure untuk mengetahui data pada table anggota berdasarkan pada 3 field yaitu angkatan, tempat lahir dan jenis kelamin
5. Definisikan stored procedure untuk memasukkan data pada table anggota
6. Definisikan stored procedure untuk mengetahui data jumlah anggota menggunakan parameter OUT
7. Definisikan stored procedure untuk mengetahui data jumlah buku berdasarkan pengarang dan tahun terbitnya menggunakan parameter INOUT
8. Source code

-1-

DELIMITER //

CREATE PROCEDURE search\_anggota()

BEGIN

SELECT \* FROM tb\_anggota;

END //

DELIMITER ;

CALL search\_anggota();

-2-

DELIMITER //

CREATE PROCEDURE search\_penerbit\_buku(IN PenerbitBuku VARCHAR(30))

BEGIN

SELECT \* FROM buku

WHERE Penerbit\_Buku = PenerbitBuku;

END //

DELIMITER ;

CALL search\_penerbit\_buku('imam');

-3-

DELIMITER //

CREATE PROCEDURE search\_tgl\_pinjam(

IN TanggalPinjam DATE,

IN KodeBuku VARCHAR (10))

BEGIN

SELECT \* FROM peminjaman

WHERE Tanggal\_Pinjam = TanggalPinjam

AND Kode\_Buku = KodeBuku;

END //

DELIMITER ;

CALL search\_tgl\_pinjam('2024-04-12','912');

-4-

DELIMITER //

CREATE PROCEDURE search\_angkatan\_anggota(

IN Angkatan VARCHAR (6),

IN TempatLahir VARCHAR (20),

IN JenisKelamin VARCHAR (15))

BEGIN

SELECT \* FROM tb\_anggota

WHERE Angkatan\_Anggota = Angkatan

AND Tempat\_Lahir\_Anggota = TempatLahir

AND Jenis\_Kelamin = JenisKelamin;

END //

DELIMITER ;

CALL search\_angkatan\_anggota('22','Sumenep','Laki-laki');

-5-

DELIMITER //

CREATE PROCEDURE input\_anggota(

idAnggota VARCHAR (10),

Nama\_Anggota VARCHAR (20),

Angkatan\_Anggota VARCHAR (6),

Tempat\_Lahir\_Anggota VARCHAR (20),

Tanggal\_Lahir\_Anggota DATE,

No\_Telp INT (12),

Jenis\_Kelamin VARCHAR (15),

Status\_Pinjam VARCHAR (15))

BEGIN

INSERT INTO tb\_anggota VALUES (idAnggota,Nama\_Anggota,Angkatan\_Anggota,Tempat\_Lahir\_Anggota,Tanggal\_Lahir\_Anggota,No\_Telp,Jenis\_Kelamin,Status\_Pinjam);

END //

DELIMITER ;

CALL input\_anggota('14', 'Akhmad Sulthon Rabbani', '22', 'Sumenep', '2003-08-04', 088291847647, 'Laki-Laki', 'Sudah');

SELECT \* FROM tb\_anggota;

-6-

DELIMITER //

CREATE PROCEDURE get\_jumlah\_anggota(

OUT jumlah\_anggota INT)

BEGIN

SELECT COUNT(\*) INTO jumlah\_anggota FROM tb\_anggota;

END //

DELIMITER ;

CALL get\_jumlah\_anggota(@total\_anggota);

SELECT @total\_anggota AS TotalAnggota;

-7-

DELIMITER //

CREATE PROCEDURE jumlah\_buku(

IN NamaPengarang VARCHAR (35),

INOUT tahunTerbit VARCHAR (10),

OUT totalBuku INT)

BEGIN

SELECT SUM(Jumlah\_Buku) INTO totalBuku

FROM buku

WHERE Pengarang\_Buku = NamaPengarang

AND Tahun\_Buku = tahunTerbit;

SET tahunTerbit = tahunTerbit + 1;

END //

DELIMITER ;

SET @tahunTerbit = '2019';

CALL jumlah\_buku('imam', @tahunTerbit , @totalBuku);

SELECT @totalBuku AS TotalBuku;

CALL input\_anggota('14', 'Akhmad Sulthon Rabbani', '22', 'Sumenep', '2003-08-04', 088291847647, 'Laki-Laki', 'Sudah');

SELECT \* FROM tb\_anggota;

-6-

DELIMITER //

CREATE PROCEDURE get\_jumlah\_anggota(

OUT jumlah\_anggota INT)

BEGIN

SELECT COUNT(\*) INTO jumlah\_anggota FROM tb\_anggota;

END //

DELIMITER ;

CALL get\_jumlah\_anggota(@total\_anggota);

SELECT @total\_anggota AS TotalAnggota;

-7-

DELIMITER //

CREATE PROCEDURE jumlah\_buku(

IN NamaPengarang VARCHAR (35),

INOUT tahunTerbit VARCHAR (10),

OUT totalBuku INT)

BEGIN

SELECT SUM(Jumlah\_Buku) INTO totalBuku

FROM buku

WHERE Pengarang\_Buku = NamaPengarang

AND Tahun\_Buku = tahunTerbit;

SET tahunTerbit = tahunTerbit + 1;

END //

DELIMITER ;

SET @tahunTerbit = '2019';

CALL jumlah\_buku('imam', @tahunTerbit , @totalBuku);

SELECT @totalBuku AS TotalBuku;

1. Hasil
2. Soal 1



1. Soal 2



1. Soal 3



1. Soal 4



1. Soal 5



1. Soal 6



1. Soal 7



1. Penjelasan

Jadi di sini saya membuat beberapa stored procedure sesuai dengan soal yang di berikan, yang Dimana Langkah pertama saya membuat databasenya lalu saya isi, setelahnya saya membuat stored procedure dengan beberapa parameter IN, OUT, INOUT. Lalu setelah saya membuat lalu saya panggil menggunakan sql CALL.

# BAB V PENUTUP

## Analisa

Penggunaan stored procedure dalam sistem manajemen basis data (DBMS) mencakup berbagai aspek, mulai dari keuntungan hingga pertimbangan yang perlu dipertimbangkan. Salah satu keuntungan utama dari penggunaan stored procedure adalah peningkatan kinerja aplikasi. Dengan menjalankan logika bisnis di dalam database, stored procedure mengurangi jumlah data yang harus dikirimkan antara server database dan aplikasi, mengurangi overhead jaringan, dan mempercepat waktu respons.

Namun, ada beberapa pertimbangan yang perlu diperhatikan dalam penggunaan stored procedure. Salah satunya adalah kompleksitas pengembangan dan pemeliharaan. Stored procedure memerlukan pengetahuan yang mendalam tentang bahasa pemrograman yang digunakan di dalam database, serta struktur data dan skema basis data yang kompleks.

## Kesimpulan

Penggunaan stored procedure dalam sistem manajemen basis data (DBMS) menawarkan sejumlah manfaat yang signifikan, termasuk peningkatan kinerja aplikasi, pengurangan lalu lintas jaringan, tingkat abstraksi yang lebih tinggi untuk pengolahan data, dan dukungan terhadap transaksi kompleks.