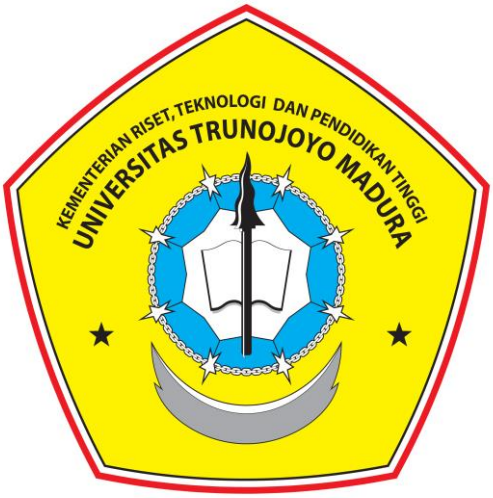
LAPORAN RESMI
MODUL III
STORED PROCEDURE



NAMA	**:**	**ANISYAFAAH**
N.R.P	**:**	**220441100105**
DOSEN	**:**	**FITRI DAMAYANTI, S.Kom., M.Kom.**
ASISTEN	**:**	**AFFAN MAULANA ZULKARNAIN**
TGL PRAKTIKUM	**:**	**26 APRIL 2024**

Disetujui : 01 Mei 2024
Asisten

AFFAN MAULANA ZULKARNAIN
20.04.411.00052

LABORATORIUM BISNIS INTELIJEN SISTEM
PRODI SISTEM INFORMASI
JURUSAN TEKNIK INFORMATIKA
FAKULTAS TEKNIK
UNIVERSITAS TRUNOJOYO MADURA

BAB I
PENDAHULUAN

1.1 Latar Belakang

Pentingnya database dalam konteks modern tak terbantahkan. Dengan semakin meningkatnya volume data yang dihasilkan oleh organisasi dan perusahaan, pengelolaan informasi secara efisien menjadi kunci keberhasilan. Database memainkan peran penting dalam menyimpan, mengelola, dan menyediakan akses terhadap data ini. Dengan menggunakan database, informasi dapat disimpan secara terstruktur, memungkinkan untuk pengambilan data yang cepat dan efisien. Salah satu pembuatan database dapat dilakukan dengan query. Dalam query kita dapat membuat sebuah procedure.

Stored procedure adalah kumpulan pernyataan SQL yang telah didefinisikan dan disimpan di dalam database. Penggunaan stored procedure memiliki beberapa keuntungan yang signifikan. Pertama, stored procedure membantu dalam meningkatkan kinerja aplikasi dengan mengurangi jumlah lalu lintas jaringan antara aplikasi dan database. Kedua, dengan menggunakan stored procedure, pengembang dapat mengelompokkan dan menyusun logika bisnis kompleks di dalam database.

Selain itu, stored procedure juga meningkatkan keamanan database dengan memungkinkan pengaturan hak akses yang lebih terperinci. Penggunaan stored procedure dapat membatasi akses langsung ke tabel-tabel di dalam database, sehingga mengurangi risiko penyalahgunaan data dan serangan keamanan. Dengan adanya kontrol akses yang lebih ketat, stored procedure membantu dalam menjaga integritas data dan mengurangi potensi kerentanan keamanan pada aplikasi yang menggunakan database.

1.2 Tujuan

- 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

2.1 Stored Procedure

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.

2.1.1 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:

```
DELIMITER //
CREATE PROCEDURE getMhs()
BEGIN
    SELECT * FROM mahasiswa;
END //
DELMITER;
```

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.

2.1.2 Aktivasi/Pemanggilan Stored Procedure:

```
<call statement> ::=

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

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

Eksekusi Query tersebut dengan memanggil procedure getMahasiswa().

```
CALL getMhs();
```

idMhs	namaMhs	almtMhs	jKlmn
1011	Ari	Surabaya	L
1012	Bagus	Malang	L
1013	Candra	Sampang	L
1014	Dhani	Surabaya	P
1018	Frinza	Kediri	P

2.1.3 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.

2.2 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, kita bisa mendapatkan semua data matakuliah di semester tertentu.

```
DELIMITER //
CREATE PROCEDURE getNamaByJenisKlamin(IN jeniskelamin varchar(1))
    BEGIN
        select * from mhs
        where jKlmn = jeniskelamin;
    END //
DELIMITER ;
```

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);
```

idMhs	namaMhs	almtMhs	jKlmn
1011	Ari	Surabaya	L
1012	Bagus	Malang	L
1013	Candra	Sampang	L

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:

```
DELIMITER //
CREATE PROCEDURE getNamaByAlmtJKelamin(
    IN Alamat VARCHAR(20),
    IN jenisKelamin VARCHAR(2))
    BEGIN
     SELECT * FROM mhs
     WHERE jKlmn = jenisKelamin
     AND almtMhs = Alamat ;
    END //
DELIMITER ;
```

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

```
CALL getNamaByAlmtJKelamin ('surabaya','L');
```

idMhs	namaMhs	almtMhs	jKlmn
1011	Ari	Surabaya	L

2.2.1 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.

```
DELIMITER //
CREATE PROCEDURE AddMahasiswa(
    IN idMhs INT(10),
    IN namaMhs VARCHAR(20),
    IN almtMhs VARCHAR(30),
    IN jKlmn VARCHAR(2))
    BEGIN
        INSERT INTO mhs
        VALUES (idMhs,namaMhs,almtMhs,jKlmn);
    END //
DELIMITER ;
```

Lakukan eksekusi terhadap procedure tersebut

```
Call AddMahasiswa('1019','Gunawan','surabaya',
'L');
```

Selanjutnya lakukan pengecekan data pada tabel mahasiswa.

```
select * from mhs; atau
select * getMhs();
```

idMhs	namaMhs	almtMhs	jKlmn
1011	Ari	Surabaya	L
1012	Bagus	Malang	L
1013	Candra	Sampang	L
1014	Dhani	Surabaya	P
1018	Frinza	Kediri	P
1019	Gunawan	surabaya	L

2.3 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 diubaholeh stored procedure.

```
DELIMITER //
CREATE PROCEDURE JumlahMahasiswa(OUT jumlah_mhs INT(3))
   BEGIN
    SELECT COUNT(idMhs)
    INTO jumlah_mhs FROM mhs;
   END //
DELIMITER ;
```

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;
```

@jumlah_mhs
6

Parameter mode OUT juga bisa dikombinasikan dengan mode IN.

2.4 INOUT

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:

```
DELIMITER //
CREATE PROCEDURE CountByGender(
    IN gender VARCHAR(2),
    OUT total INT(3))
    BEGIN
     SELECT COUNT(idMhs)
     INTO total
     FROM mhs
     WHERE jKlmn = gender;
    END //
DELIMITER ;
```

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

```
call CountByGender('P',@total);
select @total;
```

@total
2

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 III
TUGAS PENDAHULUAN

3.1 Soal

1. Apa itu stored procedure dalam konteks sistem manajemen basis data?
2. Apa perbedaan antara stored procedure dan fungsi dalam database?
3. Bagaimana query untuk membuat stored procedure? Jelaskan!
4. Apa yang dimaksud dengan parameter IN, OUT, INOUT dalam konteks stored procedure?
5. Kapan kita menggunakan parameter INOUT daripada IN atau OUT dalam stored procedure?

3.2 Jawab

1. Stored procedure adalah sebuah prosedur layaknya subprogram (subrutin) di dalam bahasa pemrograman reguler yang tersimpan di dalam katalog basis data.
2. Perbedaan utama antara stored procedure dan fungsi dalam database yaitu stored procedure dapat melakukan perubahan pada data atau struktur database, sedangkan fungsi hanya dapat digunakan untuk mengembalikan nilai berdasarkan input yang diberikan.
3. Query untuk membuat stored procedure :

```
DELIMITER //
CREATE PROCEDURE nama_prosedur()
BEGIN
    -- Perintah SQL
END //
DELIMITER ;
```

Penjelasan query :

- Perintah DELIMITER digunakan untuk mengubah deliminer standar, misalnya disini dari titik koma (;) menjadi slash ganda (//)
- Pernyataan BEGIN dan END merupakan badan (body) stored procedure, biasanya berisi perintah - perintah SQL
- DELIMITER di akhir berfungsi mengembalikan deliminer ke karakter semula.

4. Parameter IN, OUT, dan INOUT :
   - IN merupakan parameter yang mengindikasikan bahwa sebuah parameter dapat di pass ke dalam stored procedure tetapi nilainya tidak dapat diubah
   - OUT merupakan mode yang mengindikasikan bahwa stored procedure dapat mengubah parameter dan mengirimkan kembali ke program pemanggil.
   - INOUT merupakan parameter yang digunakan untuk mengirim parameter dan mendapatkan nilai kembalian yang baru.
5. Parameter INOUT digunakan ketika ingin memasukkan nilai ke dalam stored procedure dan mengembalikan nilai tersebut atau nilai yang diubah oleh stored procedure kembali ke pemanggil setelah proses selesai.

BAB IV

IMPLEMENTASI

4.1 Source Code

4.1.1 Mengisi Data Setiap Tabel

- **Tabel Petugas**

a) Source Code

```
CREATE DATABASE peminjaman;

USE peminjaman;

CREATE TABLE Petugas (
   ID_Petugas VARCHAR(10) PRIMARY KEY,
   Username VARCHAR(15) NOT NULL,
   Password VARCHAR(15) NOT NULL,
   Nama VARCHAR(25) NOT NULL
);

INSERT INTO Petugas VALUES ("P001", "Anisa01", "anisa123", "Anisyafaah");
INSERT INTO Petugas VALUES ("P002", "Icha02", "icha123", "Anisah Nuril I.");
INSERT INTO Petugas VALUES ("P003", "Pipin03", "pipin123", "Herdiyanti Fifin P.");
INSERT INTO Petugas VALUES ("P004", "Putri04", "putri123", "Firdausi Putri C.");
INSERT INTO Petugas VALUES ("P005", "Adhel05", "adhel123", "Adhelia Kusumawati");
INSERT INTO Petugas VALUES ("P006", "Nadhif06", "nadhif123", "Rayhanza Nadhif A.");
INSERT INTO Petugas VALUES ("P007", "Erick07", "erick123", "Erick Firmansyah");
INSERT INTO Petugas VALUES ("P008", "Juan08", "juan123", "Juanzha Nanda P.");
INSERT INTO Petugas VALUES ("P009", "Firman09", "firman123", "Firman Syahril");
INSERT INTO Petugas VALUES ("P010", "Akmal10", "akmal123", "Akmal Nabil H.");
```

b) Penjelasan

Kode di atas digunakan untuk membuat database peminjaman dan membuat tabel petugas serta mengisinya dengan 10 data. Untuk membuat database peminjaman menggunakan perintah CREATE TABLE peminjaman. Untuk membuat tabel petugas menggunakan perintah CREATE TABLE Petugas (namaKolom1, namaKolom2, dst.). Tabel di atas memiliki kolom yang terdiri dari id_petugas sebagai primary key, username, password, dan nama petugas. Selanjutnya untuk mengisi data pada setiap kolom menggunakan perintah INSERT INTO.

- **Tabel Buku**

a) Source Code

```
CREATE TABLE Buku (
    Kode_Buku VARCHAR(10) PRIMARY KEY,
    Judul_Buku VARCHAR(25) NOT NULL,
    Pengarang_Buku VARCHAR(30) NOT NULL,
    Penerbit_Buku VARCHAR(30) NOT NULL,
    Tahun_Buku VARCHAR(10) NOT NULL,
    Jumlah_Buku VARCHAR(5) NOT NULL,
   Status_Buku VARCHAR(10) NOT NULL,
    Klasifikasi_Buku VARCHAR(20) NOT NULL
);

INSERT INTO Buku VALUES ("B01", "Pemrograman Java Dasar", "Budi Raharjo", "Andi Offset", "2019", "15", "Tersedia", "Teknologi Informasi");
INSERT INTO Buku VALUES ("B02", "Manajemen Keuangan Modern", "Fred Weston", "Salemba Empat", "2015", "12", "Tersedia", "Keuangan");
INSERT INTO Buku VALUES ("B03", "Keajaiban Doa", "Yusuf Mansur", "Pustaka Hidayah", "2017", "10", "Tersedia", "Agama");
INSERT INTO Buku VALUES ("B04", "Kisah Inspiratif Dunia", "Yusuf Mansur", "Qanita", "2020", "11", "Tersedia", "Inspiratif");
```

```
INSERT INTO Buku VALUES ("B05", "Kimia Organik Dasar", "John McMurry", "Erlangga", "2017", "12", "Tersedia", "Kimia Organik");
INSERT INTO Buku VALUES ("B06", "Manajemen Proyek Terpadu", "Rini Setyawati", "Penerbit Andi", "2016", "10", "Tersedia", "Manajemen Proyek");
INSERT INTO Buku VALUES ("B07", "Manajemen Waktu", "David Allen", "Gramedia Pustaka Utama", "2018", "10", "Tersedia", "Pengembangan Diri");
INSERT INTO Buku VALUES ("B08", "Panduan Cepat Photoshop", "Stuart Russell", "Erlangga", "2015", "11", "Tersedia", "Kimia");
INSERT INTO Buku VALUES ("B09", "Kecerdasan Buatan", "William A. Haviland", "Erlangga", "2018", "15", "Tersedia", "Teknologi");
INSERT INTO Buku VALUES ("B10", "Jalan Menuju Sukses", "Tung Desem Waringin", "Gramedia Pustaka Utama", "2017", "15", "Tersedia", "Motivasi");
```

b) Penjelasan

Kode di atas digunakan untuk membuat tabel buku dan mengisi data sebanyak 10 data. Untuk membuat tabel buku menggunakan perintah CREATE TABLE Buku (namaKolom1, namaKolom2, dst.). Tabel di atas memiliki kolom yang terdiri dari kode buku sebagai primary key, judul buku, pengarang, penerbit, tahun terbit, status buku, dan klasifikasi buku. Selanjutnya untuk mengisi data pada setiap kolom menggunakan perintah INSERT INTO.

- **Tabel Anggota**

a) Source Code

```
CREATE TABLE Anggota (
    ID_Anggota VARCHAR(10) PRIMARY KEY,
    Nama_Anggota VARCHAR(20) NOT NULL,
    Angkatan_Anggota VARCHAR(6) NOT NULL,
    Tempat_Lahir_Anggota VARCHAR(20) NOT NULL,
    Tanggal_Lahir_Anggota DATE NOT NULL,
    No_Telp INT(12) NOT NULL,
    Jenis_Kelamin VARCHAR(15) NOT NULL,
```

```
    Status_Pinjam VARCHAR(15) NOT NULL
);

INSERT INTO Anggota VALUES ("A001", "Tasya Dwiyanti", "2022", "Bangkalan", "2004-08-15", 0858437243, "Perempuan", "Meminjam");
INSERT INTO Anggota VALUES ("A002", "Amanda Hartanto", "2022", "Surabaya", "2004-04-08", 0878909567, "Perempuan", "Meminjam");
INSERT INTO Anggota VALUES ("A003", "Denny Pranata", "2022", "Surabaya", "2004-01-01", 0821414325, "Laki-Laki", "Meminjam");
INSERT INTO Anggota VALUES ("A004", "Siti Rahayu", "2022", "Gresik", "2004-03-12", 0812304785, "Perempuan", "Meminjam");
INSERT INTO Anggota VALUES ("A005", "Andre Tanjung", "2022", "Surabaya", "2004-05-18", 0812324598, "Laki-Laki", "Meminjam");
INSERT INTO Anggota VALUES ("A006", "Maya Suryani", "2022", "Bangkalan", "2004-02-20", 0856278540, "Perempuan", "Meminjam");
INSERT INTO Anggota VALUES ("A007", "Lisa Wijaya", "2022", "Gresik", "2004-05-27", 0878794062, "Perempuan", "Meminjam");
INSERT INTO Anggota VALUES ("A008", "Johan Widodo", "2022", "Sidoarjo", "2004-04-09", 0856925670, "Laki-Laki", "Meminjam");
INSERT INTO Anggota VALUES ("A009", "Budi Santoso", "2022", "Surabaya", "2004-06-11", 0821750925, "Laki-Laki", "Meminjam");
INSERT INTO Anggota VALUES ("A010", "Lina Purnama", "2022", "Gresik", "2004-04-26", 0856278012, "Perempuan", "Meminjam");
```

b) Penjelasan

Kode di atas digunakan untuk membuat tabel anggota dan mengisi data sebanyak 10 data. Untuk membuat tabel anggota menggunakan perintah CREATE TABLE Anggota (namaKolom1, namaKolom2, dst.). Tabel di atas memiliki kolom yang terdiri dari id_anggota sebagai primary key, nama anggota, angkatan, tempat lahir, tanggal lahir, nomor telepon, jenis kelamin, dan status pinjam. Selanjutnya untuk mengisi data pada setiap kolom menggunakan perintah INSERT INTO.

- **Tabel Peminjaman**

  a) Source Code

```
CREATE TABLE Peminjaman (
    Kode_Peminjaman VARCHAR(10) NOT NULL PRIMARY KEY,
    ID_Anggota VARCHAR(10) NOT NULL,
    ID_Petugas VARCHAR(10) NOT NULL,
    Tanggal_Pinjam DATE NOT NULL,
    Tanggal_Kembali DATE NOT NULL,
   Kode_Buku VARCHAR(10) NOT NULL,
    FOREIGN KEY (ID_Petugas) REFERENCES Petugas(ID_Petugas),
    FOREIGN KEY (Kode_Buku) REFERENCES Buku(Kode_Buku),
    FOREIGN KEY (ID_Anggota) REFERENCES Anggota(ID_Anggota)
);
```

  b) Penjelasan

  Kode di atas digunakan untuk membuat tabel peminjaman dan mengisi data sebanyak 10 data. Untuk membuat tabel peminjaman menggunakan perintah CREATE TABLE Peminjaman (namaKolom1, namaKolom2, dst.). Tabel di atas memiliki kolom yang terdiri dari kode peminjaman sebagai primary key, id anggota yang terhubung dengan tabel anggota menggunakan perintah FOREIGN KEY, id petugas yang terhubung dengan tabel petugas menggunakan perintah FOREIGN KEY, tanggal pinjam, tanggal kembali, dan kode buku yang terhubung dengan tabel buku menggunakan perintah FOREIGN KEY. Selanjutnya untuk mengisi data pada setiap kolom menggunakan perintah INSERT INTO.

- **Tabel Pengembalian**

  a) Source Code

```
CREATE TABLE Pengembalian (
   Kode_Kembali VARCHAR(10) NOT NULL PRIMARY KEY,
   ID_Anggota VARCHAR(20) NOT NULL,
   Kode_Buku VARCHAR(10) NOT NULL,
   ID_Petugas VARCHAR(10) not null,
   Tanggal_Pinjam DATE NOT NULL,
   Tanggal_Kembali DATE NOT NULL,
   Denda VARCHAR(15) NOT NULL,
   FOREIGN KEY (ID_Anggota) REFERENCES
Anggota(ID_Anggota),
   FOREIGN KEY (Kode_Buku) REFERENCES Buku(Kode_Buku),
   FOREIGN KEY (ID_Petugas) REFERENCES Petugas(ID_Petugas)
);
```

  b) Penjelasan

  Kode di atas digunakan untuk membuat tabel pengembalian dan mengisi data sebanyak 10 data. Untuk membuat tabel pengembalian menggunakan perintah CREATE TABLE Pengembalian (namaKolom1, namaKolom2, dst.). Tabel di atas memiliki kolom yang terdiri dari kode kembali sebagai primary key, id anggota yang terhubung dengan tabel anggota menggunakan perintah FOREIGN KEY, kode buku yang terhubung dengan tabel buku menggunakan perintah FOREIGN KEY, id petugas yang terhubung dengan tabel petugas menggunakan perintah FOREIGN KEY, tanggal pinjam, tanggal kembali, dan denda. Selanjutnya untuk mengisi data pada setiap kolom menggunakan perintah INSERT INTO.

4.1.2 Stored Procedure Menggunakan Parameter IN Default

a) Source Code

```
-- Nomor 1
DELIMITER //
CREATE PROCEDURE Data_Petugas(Kode_Petugas VARCHAR(10))
BEGIN
    SELECT * FROM Petugas
   WHERE ID_Petugas = Kode_Petugas;
END //
DELIMITER ;

CALL Nama_Petugas("P001");
```

b) Penjelasan

Kode di atas digunakan untuk membuat sebuah prosedur yang menggunakan parameter IN Default. Kode di atas akan menampilkan data petugas berdasarkan id petugasnya. Untuk membuat prosedur menggunakan perintah CREATE PROCEDURE nama_prosedur. IN Default menjadi parameter pemanggil dimana data akan dicari berdasarkan id petugasnya sehingga kode eksekusi akan dimulai dari perintah SELECT * FROM Petugas dimana ID_Petugas = nama parameter IN. Untuk menampilkan datanya menggunakan syntax CALL nama_prosedur.

4.1.3 Stored Procedure Tabel Pengembalian Berdasarkan Kode Buku

a) Source Code

```
-- Nomor 2
DELIMITER //
CREATE PROCEDURE PengembalianBuku(Book_Code VARCHAR(10))
BEGIN
 SELECT * FROM Pengembalian
   WHERE Kode_Buku = Book_Code;
END //
DELIMITER ;

CALL PengembalianBuku("B02");
```

b) Penjelasan

Kode di atas digunakan untuk menampilkan data pengembalian berdasarkan kode bukunya. Untuk membuat prosedur menggunakan perintah CREATE PROCEDURE nama_prosedur. Parameter yang digunakan yaitu kode buku sehingga data akan dicari berdasarkan kode buku. Kode akan dimulai dari perintah SELECT * FROM Pengembalian dimana Kode_Buku = nama parameter. Untuk menampilkan datanya menggunakan syntax CALL nama_prosedur.

4.1.4 Stored Procedure Tabel Anggota Berdasarkan Angkatan dan Jenis Kelamin

a) Source Code

```
-- Nomor 3
DELIMITER //
CREATE PROCEDURE AnggotaByAktDanJK (Angkatan VARCHAR(6), JK
VARCHAR(15))
BEGIN
        SELECT * FROM Anggota
   WHERE Angkatan_Anggota = Angkatan
   AND Jenis_Kelamin LIKE CONCAT('%', JK, '%');
END //
DELIMITER ;

CALL AnggotaByAktDanJK ("2022", "Lak");
```

b) Penjelasan

Kode di atas digunakan untuk menampilkan data anggota berdasarkan angkatan dan jenis kelaminnya. Untuk membuat prosedur menggunakan perintah CREATE PROCEDURE nama_prosedur. Parameter yang digunakan yaitu angkatan dan jenis kelamin sehingga data akan dicari berdasarkan angkatan dan jenis kelamin. Kode akan dimulai dari perintah SELECT * FROM Anggota dimana Angkatan_Anggota = nama parameter pertama dan perintah Jenis_Kelamin LIKE CONCAT('%', JK, '%') dimana LIKE digunakan untuk mencocokkan pola, dan CONCAT digunakan untuk

menggabungkan string. Dalam konteks ini, '%', JK, '%' akan menghasilkan pola yang mencocokkan nilai parameter JK di mana pun di dalam nilai kolom Jenis_Kelamin.

4.1.5 Store Procedure Tabel Buku Berdasarkan Pengarang, Penerbit, dan Tahun Terbit

a) Source Code

```
-- Nomor 4
DELIMITER //
CREATE PROCEDURE BukuBy3Field(Pengarang VARCHAR(30), Penerbit
VARCHAR(30), Tahun VARCHAR(10))
BEGIN
 SELECT * FROM Buku
   WHERE Pengarang_Buku = Pengarang
   AND Penerbit_Buku = Penerbit
   AND Tahun_Buku = Tahun;
END //
DELIMITER ;

CALL BukuBy3Field ("Yusuf Mansur", "Pustaka Hidayah", "2017");
```

b) Penjelasan

Kode di atas digunakan untuk menampilkan data buku berdasarkan pengarang, penerbit, dan tahun buku. Untuk membuat prosedur menggunakan perintah CREATE PROCEDURE nama_prosedur. Parameter yang digunakan yaitu pengarang, penerbit, dan tahun sehingga data akan dicari berdasarkan pengarang, penerbit, dan tahun. Kode akan dimulai dari perintah SELECT * FROM Buku dimana Pengarang_Buku = nama parameter pertama, Penerbit_Buku = nama parameter kedua dan Tahun_Buku = nama parameter ketiga. Untuk menampilkan datanya menggunakan syntax CALL nama_prosedur.

4.1.6 Store Procedure Menambah Data Tabel Buku

a) Source Code

```
-- Nomor 5
DELIMITER //
CREATE PROCEDURE MenambahBuku(Kode VARCHAR(10), Judul VARCHAR(25), Pengarang VARCHAR(30), Penerbit VARCHAR(30), Tahun VARCHAR(10), Jumlah VARCHAR(5), StatusBuku VARCHAR(10), Klasifikasi VARCHAR(20))
BEGIN
 INSERT INTO Buku (Kode_Buku, Judul_Buku, Pengarang_Buku, Penerbit_Buku, Tahun_Buku, Jumlah_Buku, Status_Buku, Klasifikasi_Buku)
   VALUES (Kode, Judul, Pengarang, Penerbit, Tahun, Jumlah, StatusBuku, Klasifikasi);
END //
DELIMITER ;

CALL MenambahBuku ("B11", "Perjalanan Hidup", "Mario Teguh", "Gramedia Pustaka Utama", "2006", "7", "Tersedia", "Inspiratif");
```

b) Penjelasan

Kode di atas digunakan untuk menambah data buku menggunakan prosedur. Untuk membuat prosedur menggunakan perintah CREATE PROCEDURE nama_prosedur. Parameter berisi nama-nama kolom pada tabel buku sehingga terdapat delapan parameter. Kode akan dimulai dari perintah INSERT INTO Buku (nama-nama parameter) VALUES (nama-nama kolom). Untuk menambah datanya menggunakan syntax CALL nama_prosedur (isi data baru).

4.1.7 Store Procedure Data Jumlah Buku Menggunakan Parameter OUT

a) Source Code

```
-- Nomor 6
DELIMITER //
CREATE PROCEDURE JumlahBuku(OUT Jumlah VARCHAR(5))
BEGIN
  SELECT COUNT(Jumlah_Buku)
    INTO Jumlah
    FROM Buku;
END //
DELIMITER ;

CALL JumlahBuku(@Jumlah);
SELECT @Jumlah;
```

b) Penjelasan

Kode di atas digunakan untuk menampilkan jumlah buku menggunakan prosedur dengan parameter OUT. Untuk membuat prosedur menggunakan perintah CREATE PROCEDURE nama_prosedur. OUT menjadi parameter pemanggil dimana jumlah buku akan disimpan pada nama pameter OUT sehingga kode akan dimulai dari perintah SELECT COUNT(nama prosedur) INTO nama parameter OUT dari tabel Buku. Untuk menyimpan datanya menggunakan syntax CALL nama_prosedur(@ nama parameter). Untuk menampilkan datanya menggunakan syntax SELECT @ nama parameter.

4.1.8 Store Procedure Data Jumlah Anggota Berdasarkan Angkatan dan Jenis Kelamin Menggunakan Parameter INOUT

c) Source Code

```
-- Nomor 7
DELIMITER //
CREATE PROCEDURE DataAnggotaa(
   IN Angkatan VARCHAR(6),
   IN JenisKelamin VARCHAR(10),
   OUT JumlahAnggota INT
)
BEGIN
   SELECT COUNT(Nama_Anggota)
   INTO JumlahAnggota
   FROM Anggota
   WHERE Angkatan_Anggota = Angkatan
   AND Jenis_Kelamin = JenisKelamin;
END //
DELIMITER ;

CALL DataAnggotaa("2022", "Perempuan", @JumlahPerempuan);
CALL DataAnggotaa("2022", "Laki-laki", @JumlahLakiLaki);

SELECT @JumlahPerempuan, @JumlahLakiLaki;
```

d) Penjelasan

Kode di atas digunakan untuk menampilkan jumlah data anggota menggunakan prosedur berdasarkan angkatan dan jenis kelamin dengan parameter INOUT. Untuk membuat prosedur menggunakan perintah CREATE PROCEDURE nama_prosedur. IN menjadi parameter pemanggil dimana jumlah anggota akan dicari berdasarkan pameter IN (Angkatan). IN menjadi parameter pemanggil dimana jumlah anggota akan dicari berdasarkan pameter IN (Jenis Kelamin). OUT menjadi parameter pemanggil dimana jumlah buku akan disimpan pada nama pameter OUT sehingga kode akan dimulai dari perintah SELECT COUNT(nama prosedur) INTO nama parameter OUT dari tabel Anggota dimana Angkatan_Anggota = nama

parameter IN pertama dan Jenis_Kelamin = nama parameter IN kedua. Untuk menyimpan datanya menggunakan syntax CALL nama_prosedur(“”, “”, @ nama parameter). Untuk menampilkan datanya menggunakan syntax SELECT @ nama parameter.

4.2 Hasil

4.2.1 Data Setiap Tabel

- **Tabel Petugas**

ID_Petugas	Username	Password	Nama
P001	Anisa01	anisa123	Anisyafaah
P002	Icha02	icha123	Anisah Nuril I.
P003	Pipin03	pipin123	Herdiyanti Fifin P.
P004	Putri04	putri123	Firdausi Putri C.
P005	Adhel05	adhel123	Adhelia Kusumawati
P006	Nadhif06	nadhif123	Rayhanza Nadhif A.
P007	Erick07	erick123	Erick Firmansyah
P008	Juan08	juan123	Juanzha Nanda P.
P009	Firman09	firman123	Firman Syahril
P010	Akmal10	akmal123	Akmal Nabil H.
NULL	NULL	NULL	NULL

- **Tabel Buku**

Kode_Buku	Judul_Buku	Pengarang_Buku	Penerbit_Buku	Tahun_Buku	Jumlah_Buku	Status_Buku	Klasifikasi_Buku
B01	Pemrograman Java Dasar	Budi Raharjo	Andi Offset	2019	15	Tersedia	Teknologi Informasi
B02	Manajemen Keuangan Modern	Fred Weston	Salemba Empat	2015	12	Tersedia	Keuangan
B03	Keajaiban Doa	Yusuf Mansur	Pustaka Hidayah	2017	10	Tersedia	Agama
B04	Kisah Inspiratif Dunia	Yusuf Mansur	Qanita	2020	11	Tersedia	Inspiratif
B05	Kimia Organik Dasar	John McMurry	Erlangga	2017	12	Tersedia	Kimia Organik
B06	Manajemen Proyek Terpadu	Rini Setyawati	Penerbit Andi	2016	10	Tersedia	Manajemen Proyek
B07	Manajemen Waktu	David Allen	Gramedia Pustaka Utama	2018	10	Tersedia	Pengembangan Diri
B08	Panduan Cepat Photoshop	Stuart Russell	Erlangga	2015	11	Tersedia	Kimia
B09	Kecerdasan Buatan	William A. Haviland	Erlangga	2018	15	Tersedia	Teknologi
B10	Jalan Menuju Sukses	Tung Desem Waringin	Gramedia Pustaka Utama	2017	15	Tersedia	Motivasi
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

- **Tabel Anggota**

ID_Anggota	Nama_Anggota	Angkatan_Anggota	Tempat_Lahir_Anggota	Tanggal_Lahir_Anggota	No_Telp	Jenis_Kelamin	Status_Pinjam
A001	Tasya Dwiyanti	2022	Bangkalan	2004-08-15	858437243	Perempuan	Meminjam
A002	Amanda Hartanto	2022	Surabaya	2004-04-08	878909567	Perempuan	Meminjam
A003	Denny Pranata	2022	Surabaya	2004-01-01	821414325	Laki-Laki	Meminjam
A004	Siti Rahayu	2022	Gresik	2004-03-12	812304785	Perempuan	Meminjam
A005	Andre Tanjung	2022	Surabaya	2004-05-18	812324598	Laki-Laki	Meminjam
A006	Maya Suryani	2022	Bangkalan	2004-02-20	856278540	Perempuan	Meminjam
A007	Lisa Wijaya	2022	Gresik	2004-05-27	878794062	Perempuan	Meminjam
A008	Johan Widodo	2022	Sidoarjo	2004-04-09	856925670	Laki-Laki	Meminjam
A009	Budi Santoso	2022	Surabaya	2004-06-11	821750925	Laki-Laki	Meminjam
A010	Lina Purnama	2022	Gresik	2004-04-26	856278012	Perempuan	Meminjam
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

- **Tabel Peminjaman**

Kode_Peminjaman	ID_Anggota	ID_Petugas	Tanggal_Pinjam	Tanggal_Kembali	Kode_Buku
PI01	A001	P001	2024-01-12	2024-01-18	B01
PI02	A002	P002	2024-01-20	2024-01-25	B02
PI03	A003	P003	2024-02-01	2024-02-06	B03
PI04	A004	P004	2024-02-09	2024-02-14	B04
PI05	A005	P005	2024-02-17	2024-02-22	B05
PI06	A006	P006	2024-03-10	2024-03-15	B06
PI07	A007	P007	2024-03-17	2024-03-22	B07
PI08	A008	P008	2024-03-21	2024-03-26	B08
PI09	A009	P009	2024-04-20	2024-04-25	B09
PI10	A010	P010	2024-04-22	2024-04-27	B10
NULL	NULL	NULL	NULL	NULL	NULL

- **Tabel Pengembalian**

Kode_Kembali	ID_Anggota	Kode_Buku	ID_Petugas	Tanggal_Pinjam	Tanggal_Kembali	Denda
K01	A001	B01	P001	2024-01-12	2024-01-15	Tidak Didenda
K02	A002	B02	P002	2024-01-20	2024-01-25	Tidak Didenda
K03	A003	B03	P003	2024-02-01	2024-02-07	Rp 3000
K04	A004	B04	P004	2024-02-09	2024-02-17	Rp 6000
K05	A005	B05	P005	2024-02-17	2024-02-20	Tidak Didenda
K06	A006	B06	P006	2024-03-10	2024-03-16	Rp 3000
K07	A007	B07	P007	2024-03-17	2024-03-19	Tidak Didenda
K08	A008	B08	P008	2024-03-21	2024-03-25	Tidak Didenda
K09	A009	B09	P009	2024-04-20	2024-04-21	Tidak Didenda
K10	A010	B10	P010	2024-04-22	2024-04-29	Rp 6000
NULL	NULL	NULL	NULL	NULL	NULL	NULL

4.2.2 Stored Procedure Menggunakan Parameter IN Default

ID_Petugas	Username	Password	Nama
P001	Anisa01	anisa123	Anisyafaah

4.2.3 Stored Procedure Tabel Pengembalian Berdasarkan Kode Buku

Kode_Kembali	ID_Anggota	Kode_Buku	ID_Petugas	Tanggal_Pinjam	Tanggal_Kembali	Denda
K02	A002	B02	P002	2024-01-20	2024-01-25	Tidak Didenda

4.2.4 Stored Procedure Tabel Anggota Berdasarkan Angkatan dan Jenis Kelamin

ID_Anggota	Nama_Anggota	Angkatan_Anggota	Tempat_Lahir_Anggota	Tanggal_Lahir_Anggota	No_Telp	Jenis_Kelamin	Status_Pinjam
A003	Denny Pranata	2022	Surabaya	2004-01-01	821414325	Laki-Laki	Meminjam
A005	Andre Tanjung	2022	Surabaya	2004-05-18	812324598	Laki-Laki	Meminjam
A008	Johan Widodo	2022	Sidoarjo	2004-04-09	856925670	Laki-Laki	Meminjam
A009	Budi Santoso	2022	Surabaya	2004-06-11	821750925	Laki-Laki	Meminjam

4.2.5 Stored Procedure Tabel Buku Berdasarkan Pengarang, Penerbit, dan Tahun Terbit

Kode_Buku	Judul_Buku	Pengarang_Buku	Penerbit_Buku	Tahun_Buku	Jumlah_Buku	Status_Buku	Klasifikasi_Buku
B03	Keajaiban Doa	Yusuf Mansur	Pustaka Hidayah	2017	10	Tersedia	Agama

4.2.6 Stored Procedure Menambah Data Tabel Buku

Kode_Buku	Judul_Buku	Pengarang_Buku	Penerbit_Buku	Tahun_Buku	Jumlah_Buku	Status_Buku	Klasifikasi_Buku
B01	Pemrograman Java Dasar	Budi Raharjo	Andi Offset	2019	15	Tersedia	Teknologi Informasi
B02	Manajemen Keuangan Modern	Fred Weston	Salemba Empat	2015	12	Tersedia	Keuangan
B03	Keajaiban Doa	Yusuf Mansur	Pustaka Hidayah	2017	10	Tersedia	Agama
B04	Kisah Inspiratif Dunia	Yusuf Mansur	Qanita	2020	11	Tersedia	Inspiratif
B05	Kimia Organik Dasar	John McMurry	Erlangga	2017	12	Tersedia	Kimia Organik
B06	Manajemen Proyek Terpadu	Rini Setyawati	Penerbit Andi	2016	10	Tersedia	Manajemen Proyek
B07	Manajemen Waktu	David Allen	Gramedia Pustaka Utama	2018	10	Tersedia	Pengembangan Diri
B08	Panduan Cepat Photoshop	Stuart Russell	Erlangga	2015	11	Tersedia	Kimia
B09	Kecerdasan Buatan	William A. Haviland	Erlangga	2018	15	Tersedia	Teknologi
B10	Jalan Menuju Sukses	Tung Desem Waringin	Gramedia Pustaka Utama	2017	15	Tersedia	Motivasi
B11	Perjalanan Hidup	Mario Teguh	Gramedia Pustaka Utama	2006	7	Tersedia	Inspiratif
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

4.2.7 Stored Procedure Data Jumlah Buku Menggunakan Parameter OUT

@Jumlah
11

4.2.8 Stored Procedure Data Jumlah Anggota Berdasarkan Angkatan dan Jenis Kelamin Menggunakan Parameter INOUT

@JumlahPerempuan	@JumlahLakiLaki
6	4

BAB V
PENUTUP

5.1 Analisa

Dari hasil praktikum, praktikan menganalisa bahwa penggunaan stored procedure dalam basis data telah menjadi sebuah praktik umum yang memainkan peran kunci dalam pengembangan perangkat lunak yang kompleks. Salah satu manfaat utama dari penggunaan stored procedure adalah efisiensi dalam kinerja dan pengelolaan data. Dengan menyimpan logika bisnis di dalam database, pengguna dapat mengurangi lalu lintas jaringan dengan menghindari pengiriman kueri kompleks secara terpisah dari aplikasi ke basis data. Hal ini juga meningkatkan keamanan data dengan mengontrol akses langsung ke tabel, karena pengguna hanya dapat berinteraksi dengan database melalui stored procedure yang telah ditentukan oleh administrator database.

Namun, penggunaan stored procedure juga memiliki beberapa kelemahan. Salah satunya adalah kompleksitas pengelolaan dan pemeliharaannya. Stored procedure yang kompleks dan besar bisa sulit untuk dipahami dan dikelola, terutama ketika tim pengembang berubah atau ketika dilakukan pembaruan struktur basis data. Selain itu, ketergantungan pada stored procedure bisa menghambat portabilitas aplikasi, karena beberapa platform basis data memiliki perbedaan sintaksis dan fitur dalam implementasi stored procedure mereka.

5.2 Kesimpulan

1. Stored Procedure adalah sebuah prosedur layaknya subprogram (subrutin) di dalam bahasa pemrograman reguler yang tersimpan di dalam katalog basis data.
2. Dalam Implementasinya, penggunaan stored procedure sering melibatkan parameter. Di MySQL, parameter stored procedure dibedakan menjadi tiga mode: IN, OUT, dan INOUT.
3. Dalam database, "CALL" digunakan untuk menjalankan stored procedure, sementara "SELECT @parameter_name" digunakan untuk mendapatkan nilai dari parameter atau variabel setelah eksekusi stored procedure selesai.