LAPORAN RESMI

MODUL V

TYPE DATA/STORED PROCEDURE

NAMA	**:**	**ANISYAFAAH**
N.R.P	**:**	**220441100105**
DOSEN	**:**	**FITRI DAMAYANTI, S.Kom., M.Kom.**
ASISTEN	**:**	**AFFAN MAULANA ZULKARNAIN**
TGL PRAKTIKUM	**:**	**08 MEI 2024**

Disetujui : 14 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

Di era digital yang semakin berkembang pesat ini, pemahaman tentang tipe data dan penggunaan stored procedure dalam basis data menjadi semakin penting. Pertama-tama, pemahaman yang kuat tentang tipe data memungkinkan kita untuk mengelola informasi dengan lebih efisien dan efektif. Dengan memahami karakteristik setiap tipe data, seperti string, integer, atau float, kita dapat mengoptimalkan struktur basis data untuk menyimpan informasi dengan benar, menghindari pemborosan ruang penyimpanan, dan meningkatkan kinerja sistem secara keseluruhan.

Kedua, penggunaan stored procedure menjadi kunci dalam meningkatkan kinerja dan keamanan basis data. Stored procedure adalah sekumpulan pernyataan SQL yang telah diproses dan disimpan dalam basis data untuk dieksekusi ulang. Dengan menggunakan stored procedure, kita dapat menghindari pengulangan kode, mengurangi beban jaringan dengan mengirimkan hanya panggilan stored procedure, serta meningkatkan keamanan data dengan mengontrol akses langsung ke tabel-tabel basis data.

Terakhir, dalam era di mana data menjadi aset utama bagi banyak organisasi, pemahaman tentang tipe data dan penggunaan stored procedure menjadi landasan untuk inovasi dan pengembangan aplikasi yang handal. Dengan menerapkan prinsip-prinsip ini, para pengembang dapat membuat aplikasi yang lebih skalabel, mudah dielola, dan dapat diandalkan, sehingga memungkinkan organisasi untuk memanfaatkan potensi penuh dari sumber daya informasi yang mereka miliki.

1.2 Tujuan

- Mampu memahami dan membuat stored procedure pada basis data
- Mampu menggunakan tipe dta sesuai kebutuhan pada stored procedure

BAB II
DASAR TEORI

2.1 Tipe Data

Dalam Database Data Type adalah suatu fungsi (function) yang digunakan untuk mengidentifikasi batasan suatu kolom dalam menyimpan dan penulisan format suatu data atau konten tertentu. Penggunaan typedata pada database memiliki beberapa fungsi yaitu: Untuk memberikan batasan atau format pada kolom table suatu database.

Ada lima jenis tipe data sesuai dengan SQL - ANSI 1993 yaitu character string, numeric, temporal, binary, dan boolean.

2.1.1 Character String

Atribut seperti nama dan alamat direpresentasikan oleh character string. Ada2 macam tipe data untuk merepresentasikan character string, yaitu:

- CHARACTER(<panjang>)

  CHAR(<panjang>) menspesifikasikan karakter dengan panjang yang tetap. Sisa karakter yang tidak terpakai umumnya digantikan oleh padding characters (spasi).

- CHARACTER VARYING(<panjang>)

  Atau VARCHAR(<panjang>) menspesifikasikan karakter dengan panjang yang fleksibel dan maksimum sesuai dengan <panjang>.

  Sintak :

  Type Data [(M)]

  Misal**:**

  CHAR [(M)]

  VARCHAR [(M)]

```
CREATE TABLE contoh_cha (cha CHAR(5), varcha VARCHAR(5));
INSERT INTO contoh_cha values ('a  ','a  ');
INSERT INTO contoh_cha values ('dunia','dunia');
INSERT INTO contoh_cha VALUES ('basisdata','basisdata');
```

2.1.2 Numeric

Data-data seperti usia dan gaji disimpan dalam bentuk angka. Penyimpanan dalam bentuk angka menggunakan tipe data numeric. Ada empat macam tipe data numeric, yaitu:

Numeric, dari namanya sudah pasti numeric berarti digunakan pada kolom yang menyimpan data berupa angka. Tipe Data numeric memiliki beberpa format penulisan mislakan bilangan desimal, bilangan bulat, dll. Berikut ini beberapa contoh format dari tipe data numeric :

Tabel Tipe Data Numeric (Angka)

Tipe Data	Fungsi	Jangkauan / Range
INT	Menyimpan data dalam bentuk *Interger* atau **bilangan bulat** dapat bernilai positif atau negatif.	-2147483648 s/d 2147483647
TINYINT		-128 s/d 127
SMALLINT		-32.768 s/d 32.767
MEDIUMINT		-8.388.608 s/d 8.388.607
BIGINT		-9223372036854775808 s/d 9223372036854775807
FLOAT	Menyimpan data bilangan pecahan positif atau negatif	3.402823466E+38 s/d -1.175494351E-38, 0, dan 1.175494351E-38 s/d 3.402823466E+38.
DOUBLE		-1.79...E+308 s/d -2.22...E-308, 0, dan 2.22...E-308 s/d 1.79...E+308
DECIMAL / NUMERIC		-1.79...E+308 s/d -2.22...E-308, 0, dan 2.22...E-308 s/d 1.79...E+308

Sintak:

Type Data [(M[,D])] [UNSIGNED] [ZEROFILL]

Misal :

INT/DECIMAL/FLOAT/DOUBLE[(M)] [UNSIGNED] [ZEROFILL]

Keterangan :

- **M** : menunjukkan lebar karakter maksimum, jumlah digit keseluruhan
- **D** : jumlah digit dibelakang koma
- **tanda [dan]** berarti pemakaiannya adalah optional

Contoh :

```
CREATE TABLE contoh_int (mini TINYINT, kecil SMALLINT UNSIGNED,sedang MEDIUMINT(4) ZEROFILL, biasa INT(4) UNSIGNED,besar BIGINT(6) UNSIGNED ZEROFILL);
```

2.1.3 Menghapus Stored Procedure:

Temporal merupakan tipe data yang menyimpan tanggal dan waktu yang disesuikan dengan system-timezone (komputer). Sebagai contoh data temporal adalah data tentang tanggal lahir. Ada dua macam tipe data temporal, yaitu:

- DATETIME. Tipe data ini menyimpan informasi tanggal, waktu atau bahkan keduanya. Dalam SQL Server, tipe data ini menyimpan dengan tingkat akurasi sampai 3,33 milidetik. Sedangkan untuk SMALLDATETIME hanya sampai 1 menit. Dalam tipe data ini, juga terdapat tipe data TIMESTAMP dengan tingkat akurasi sampai dengan 9 digit.
- INTERVAL. Umumnya digunakan untuk menyimpan periode seperti garansi. Ada 2 macam yaitu (1) YEAR-MONTH dan (2) DAY-TIME. SQL Server tidak mempunyai tipe data ini.

Tipe Data	Jangkauan	Ukuran	Zero Value
DATE	□1000-01-01□ to □9999-12-31□	3 byte	□0000-00-00□
DATETIME	□1000-□01-01 00:00:01□ to □9999-12-31 23:59:59□	8 byte	□0000-00-00 00:00:00□
TIMESTAMP	□1970-01-01 00:00:00□ to □2038-01-18 22:14:07□	4 byte	□0000-00-00 00:00:00□
TIME	□□838:59:59□ to □838:59:58□	3 byte	□00:00:00□
YEAR(2)	00 to 99	1 byte	□00□
YEAR(4)	1901 to 2155	1 byte	□0000□

Sintak : type data;

Contohnya :

```
CREATE TABLE contoh_date (dat DATE, tim TIME, dattim
DATETIME,timestam TIMESTAMP, yea YEAR);
```

Atau

```
SELECT NOW(), CURDATE(), CURTIME();
```

2.2 Sintaks Stored Procedure

```
<create procedure statement> ::=
CREATE PROCEDURE <procedure name> ( [ <parameter list>
] )
<routine body>
<parameter list> ::=
<parameter specification>
[ , <parameterspecification> ]...
<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>
```

2.2.1 Aktivasi/Pemanggilan Stored Procedure

```
<call statement> ::=
CALL [ <database name> . ] <stored procedure name>
( [ <scalar expression> [ , <scalar expression> ]...
] )
```

2.2.2 Menghapus Stored Procedure

```
<drop procedure statement> ::
=DROP PROCEDURE [ IF EXISTS ]
[ <database name> . ] <procedure name>
```

Pernyataan pembuatan stored procedure :

```
DELIMITER $$
CREATE PROCEDURE set_counter(INOUT count INT(4),IN inc
INT(4))BEGIN
SET count = count + inc;
END$$ DELIMITER ;
```

Cara memanggilnya dengan mengset isi awal :

```
SET @counter = 1;
CALL set_counter(@counter,1); -- 2
CALL set_counter(@counter,2); -- 3
CALL set_counter(@counter,9); -- 12
SELECT @counter; -- 12
```

BAB III

TUGAS PENDAHULUAN

3.1 Soal

1. Jelaskan perbedaan antara tipe data INTEGER dan FLOAT dalam basis data. Berikan contoh bagaimana masing-masing tipe data tersebut dapat digunakan dalam sebuah tabel database.
2. Jelaskan perbedaan antara tipe data CHAR dan VARCHAR dalam database. Dalam kondisi apa Anda akan memilih menggunakan CHAR dibandingkan VARCHAR?
3. Apa perbedaan antara tipe data INT, BIGINT, dan SMALLINT dalam basis data? Berikan contoh kasus penggunaan masing-masing tipe data tersebut.
4. Bagaimana tipe data DATE, TIME, dan DATETIME berbeda dalam cara mereka menyimpan data? Berikan contoh situasi di mana Anda akan menggunakan masing-masing tipe data tersebut.

3.2 Jawab

1. Perbedaan utama antara tipe data INTEGER dan FLOAT yaitu jika INTEGER digunakan untuk menyimpan data berupa bilangan bulat sedangkan FLOAT untuk menyimpan data berupa bilangan desimal. Contoh:

   a) INTEGER

```
CREATE TABLE Produk (
    id_produk INT(11) NOT NULL PRIMARY KEY,
    nama_produk VARCHAR(50)
);
```

   Penjelasan:

   - id_produk : untuk menyimpan ID unik produk (bilangan bulat)
   - nama_produk : untuk menyimpan nama produk

   b) FLOAT

```
CREAT TABLE Transaksi (
    id_transaksi INT(11) NOT NULL PRIMARY KEY,
    total_harga FLOAT
);
```

Penjelasan :

- id_transaksi : untuk menyimpan ID unik setiap transaksi
- total_harga : untuk menyimpan total harga dalam desimal

2. Perbedaan utama antara tipe data CHAR dan VARCHAR yaitu jika CHAR digunakan untuk menspesifikasikan karakter dengan panjang tetap sedangkan VARCHAR untuk menspesifikasikan karakter dengan panjang yang fleksibel dan bervariasi.
   CHAR lebih cocok digunakan ketika panjang string yang disimpan relatif tetap dan konsisten diseluruh baris tabel, seperti kode pos, kode produk, atau inisial nama negara.
3. Perbedaan antara tipe data INT, BIGINT, dan SMALLINT dalam basis data terletak pada rentang nilai yang dapat disimpan dan ukuran penyimpanan yang digunakan. Rentang nilai pada masing-masing tipe data tersebut antara lain :
   a) INT (-2147483648 hingga 2147483647)
      Contoh kasus :
      - Kolom yang menyimpan jumlah produk dalam stok suatu toko
   b) BIGINT (-9223372036854775808 hingga 9223372036854775807)
      Contoh kasus :
      - Kolom yang menyimpan ID transaksi dalam sistem e-commerce besar
   c) SMALLINT (-32.768 hingga 32.767)
      Contoh kasus :
      - Kolom yang menyimpan jumlah anak dalam sebuah keluarga
4. Perbedaan utama antara tipe data DATE, TIME, DATETIME terletak pada cara mereka menyimpan data (format data). Format data pada masing-masing tipe data tersebut antara lain :
   a) DATE (YYYY-MM-DD)
      Contohnya yaitu tanggal transaksi dalam tabel penjualan
   b) TIME (HH:MM:SS)
      Contohnya yaitu waktu mulai dan waktu selesai dalam tabel acara
   c) DATETIME (YYYY-MM-DD HH:MM:SS)
      Contohnya yaitu catatan waktu dan tanggal terjadinya suatu aktivitas pada tabel log aktivitas.

BAB IV
IMPLEMENTASI

4.1 Source Code

4.2.1 Mengisi Data Setiap Tabel

- **Tabel Produk**

a) Source Code

```
CREATE DATABASE Logistik_Pakaian;

USE Logistik_Pakaian;

CREATE TABLE Produk (
ID_Produk INT(11) NOT NULL PRIMARY KEY,
   Nama_Produk VARCHAR(100),
   Kategori_Produk VARCHAR(50),
   Harga DOUBLE,
   Berat FLOAT
);

INSERT INTO Produk (ID_Produk, Nama_Produk, Kategori_Produk, Harga, Berat) VALUES
        (1, 'Kemeja Putih Polos', 'Kemeja', 150000, 0.3),
        (2, 'Celana Jeans Slim Fit', 'Celana', 250000, 0.5),
        (3, 'Gaun Floral Maxi', 'Gaun', 350000, 0.7),
        (4, 'Kaos Oblong Basic', 'Kaos', 80000, 0.2),
        (5, 'Jaket Denim Washed', 'Jaket', 300000, 0.6),
        (6, 'Rok Midi A-Line', 'Rok', 120000, 0.4),
        (7, 'Hoodie Sweater', 'Sweater', 180000, 0.5),
        (8, 'Kemeja Tartan Flannel', 'Kemeja', 200000, 0.4),
        (9, 'Celana Panjang Chino', 'Celana', 180000, 0.4),
        (10, 'Blazer Formal', 'Blazer', 400000, 0.8);
```

b) Penjelasan

Kode di atas digunakan untuk membuat database Logistik_Pakaian dan membuat tabel produk serta mengisinya dengan 10 data. Untuk membuat database logistik pakaian menggunakan perintah CREATE DATABASE Logistik_Pakaian. Untuk membuat tabel produk menggunakan

perintah CREATE TABLE Produk (namaKolom1, namaKolom2, dst.). Tabel di atas memiliki kolom yang terdiri dari id produk sebagai primary key, nama produk, kategori produk, harga, dan berat. Selanjutnya untuk mengisi data pada setiap kolom menggunakan perintah INSERT INTO.

- **Tabel Supplier**

  a) Source Code

```
CREATE TABLE Supplier (
ID_Supplier INT(11) NOT NULL PRIMARY KEY,
   Nama_Supplier VARCHAR(100),
   Alamat VARCHAR(255),
   Telepon CHAR(15),
   Email VARCHAR(100)
);

INSERT INTO Supplier (ID_Supplier, Nama_Supplier, Alamat, Telepon, Email) VALUES
        (1, 'PT. Amanah Textile', 'Jl. Pahlawan No. 123, Jakarta', '021-12345678', 'info@amanah-textile.com'),
        (2, 'CV. Maju Jaya Garment', 'Jl. Raya Industri No. 45, Bandung', '022-98765432', 'info@maju-jaya-garment.com'),
        (3, 'UD. Bersama Fashion', 'Jl. Merdeka No. 87, Surabaya', '031-56789012', 'info@bersama-fashion.co.id'),
        (4, 'PT. Sentosa Apparel', 'Jl. Pelangi No. 55, Semarang', '024-34567890', 'info@sentosa-apparel.com'),
        (5, 'CV. Makmur Abadi Textile', 'Jl. Jaya No. 10, Solo', '0271-2345678', 'info@makmur-abadi-textile.co.id'),
        (6, 'UD. Sejahtera Garment', 'Jl. Damai No. 20, Medan', '061-7890123', 'info@sejahtera-garment.com'),
        (7, 'PT. Lancar Jaya Fashion', 'Jl. Harmoni No. 30, Malang', '0341-8901234', 'info@lancar-jaya-fashion.com'),
        (8, 'CV. Bahagia Textile', 'Jl. Bahagia No. 5, Makassar', '0411-5678901', 'info@bahagia-textile.co.id'),
        (9, 'PT. Jaya Bersama Apparel', 'Jl. Bersama No. 3, Palembang', '0711-2345678', 'info@jaya-bersama-apparel.com'),
```

```
        (10, 'UD. Sukses Fashion', 'Jl. Sukses No. 8, Yogyakarta', '0274-
7890123', 'info@sukses-fashion.co.id');
```

b) Penjelasan

Kode di atas digunakan untuk membuat tabel supplier dan mengisi data sebanyak 10 data. Untuk membuat tabel supplier menggunakan perintah CREATE TABLE Supplier (namaKolom1, namaKolom2, dst.). Tabel di atas memiliki kolom yang terdiri dari id supplier sebagai primary key, nama supplier, alamat, telepon, dan email supplier. Selanjutnya untuk mengisi data pada setiap kolom menggunakan perintah INSERT INTO.

- **Tabel Gudang**

a) Source Code

```
CREATE TABLE Gudang (
ID_Gudang INT(11) NOT NULL PRIMARY KEY,
   Nama VARCHAR(100),
   Alamat VARCHAR(255)
);

INSERT INTO Gudang (ID_Gudang, Nama, Alamat) VALUES
        (1, 'Gudang Utama', 'Jl. Gatot Subroto No. 123, Jakarta'),
        (2, 'Gudang Pusat', 'Jl. Sudirman No. 45, Bandung'),
        (3, 'Gudang Sentral', 'Jl. Diponegoro No. 87, Surabaya'),
        (4, 'Gudang Logistik', 'Jl. Gajah Mada No. 55, Semarang'),
        (5, 'Gudang Distribusi', 'Jl. Pemuda No. 10, Solo'),
        (6, 'Gudang Amanah', 'Jl. Merdeka No. 20, Medan'),
        (7, 'Gudang Maju', 'Jl. Dipa No. 30, Malang'),
        (8, 'Gudang Bersama', 'Jl. Hasanuddin No. 5, Makassar'),
        (9, 'Gudang Sejahtera', 'Jl. Kapten No. 3, Palembang'),
        (10, 'Gudang Lancar', 'Jl. Pahlawan No. 8, Yogyakarta');
```

b) Penjelasan

Kode di atas digunakan untuk membuat tabel gudang dan mengisi data sebanyak 10 data. Untuk membuat tabel gudang

menggunakan perintah CREATE TABLE Gudang (namaKolom1, namaKolom2, dst.). Tabel di atas memiliki kolom yang terdiri dari id gudang sebagai primary key, nama gudang, dan alamat. Selanjutnya untuk mengisi data pada setiap kolom menggunakan perintah INSERT INTO.

- **Tabel Stok**

  a) Source Code

```
CREATE TABLE Stok (
ID_Stok INT(11) NOT NULL PRIMARY KEY,
   ID_Produk INT(11),
   ID_Gudang INT(11),
   Jumlah INT(11),
   Tanggal_Update DATETIME,
   FOREIGN KEY (ID_Produk) REFERENCES Produk (ID_Produk),
   FOREIGN KEY (ID_Gudang) REFERENCES Gudang (ID_Gudang)
);

INSERT INTO Stok (ID_Stok, ID_Produk, ID_Gudang, Jumlah, Tanggal_Update) VALUES
        (1, 1, 1, 100, '2024-05-01 09:00:00'),
        (2, 2, 2, 80, '2024-05-02 09:15:00'),
        (3, 3, 3, 50, '2024-05-03 09:30:00'),
        (4, 4, 4, 120, '2024-05-04 09:45:00'),
        (5, 5, 5, 70, '2024-05-05 10:00:00'),
        (6, 6, 6, 90, '2024-05-06 10:15:00'),
        (7, 7, 7, 60, '2024-05-07 10:30:00'),
        (8, 8, 8, 110, '2024-05-08 10:45:00'),
        (9, 9, 9, 85, '2024-05-09 11:00:00'),
        (10, 10, 10, 40, '2024-05-10 11:15:00');
```

  b) Penjelasan

  Kode di atas digunakan untuk membuat tabel stok dan mengisi data sebanyak 10 data. Untuk membuat tabel stok menggunakan perintah CREATE TABLE Stok (namaKolom1, namaKolom2, dst.). Tabel di atas memiliki kolom yang terdiri dari id stok

sebagai primary key, id produk yang terhubung dengan tabel produk menggunakan perintah FOREIGN KEY, id gudang yang terhubung dengan tabel gudang menggunakan perintah FOREIGN KEY, jumlah, dan tanggal update. Selanjutnya untuk mengisi data pada setiap kolom menggunakan perintah INSERT INTO.

- **Tabel Karyawan**

  a) Source Code

```
CREATE TABLE Karyawan (
ID_Karyawan INT(11) NOT NULL PRIMARY KEY,
   ID_Gudang INT(11),
   Nama VARCHAR(100),
   Alamat VARCHAR(255),
   Posisi VARCHAR(50),
   Gaji DOUBLE,
   FOREIGN KEY (ID_Gudang) REFERENCES Gudang (ID_Gudang)
);

INSERT INTO Karyawan (ID_Karyawan, ID_Gudang, Nama, Alamat, Posisi, Gaji) VALUES
(1, 1, 'Budi Santoso', 'Jl. Merdeka No. 1, Jakarta', 'Manajer Gudang', 8000000),
(2, 2, 'Ani Wijaya', 'Jl. Sudirman No. 2, Bandung', 'Staf Gudang', 5000000),
(3, 3, 'Cahyo Nugroho', 'Jl. Diponegoro No. 3, Surabaya', 'Staf Gudang', 5000000),
(4, 4, 'Dewi Kurniawan', 'Jl. Gajah Mada No. 4, Semarang', 'Staf Gudang', 5000000),
(5, 5, 'Eka Setiawan', 'Jl. Pemuda No. 5, Solo', 'Staf Gudang', 5000000),
(6, 6, 'Fita Dewanti', 'Jl. Merdeka No. 6, Medan', 'Staf Gudang', 5000000),
(7, 7, 'Galih Susanto', 'Jl. Dipa No. 7, Malang', 'Staf Gudang', 5000000),
(8, 8, 'Hani Maulana', 'Jl. Hasanuddin No. 8, Makassar', 'Staf Gudang', 5000000),
(9, 9, 'Indra Wibowo', 'Jl. Kapten No. 9, Palembang', 'Staf Gudang', 5000000),
```

```
(10, 10, 'Joko Prasetyo', 'Jl. Pahlawan No. 10, Yogyakarta', 'Staf Gudang', 5000000);
```

b) Penjelasan

Kode di atas digunakan untuk membuat tabel karyawan dan mengisi data sebanyak 10 data. Untuk membuat tabel karyawan menggunakan perintah CREATE TABLE Karyawan (namaKolom1, namaKolom2, dst.). Tabel di atas memiliki kolom yang terdiri dari id karyawan sebagai primary key, id gudang yang terhubung dengan tabel gudang menggunakan perintah FOREIGN KEY, nama karyawan, alamat, posisi, dan gaji. Selanjutnya untuk mengisi data pada setiap kolom menggunakan perintah INSERT INTO.

- **Tabel Transaksi**

a) Source Code

```
CREATE TABLE Transaksi (
ID_Transaksi INT(11) NOT NULL PRIMARY KEY,
   ID_Produk INT(11),
   ID_Supplier INT(11),
   ID_Karyawan INT(11),
   Jumlah INT(11),
   Total_Harga DOUBLE,
   Tanggal_Transaksi DATETIME,
   FOREIGN KEY (ID_Produk) REFERENCES Produk (ID_Produk),
   FOREIGN KEY (ID_Supplier) REFERENCES Supplier (ID_Supplier),
   FOREIGN KEY (ID_Karyawan) REFERENCES Karyawan (ID_Karyawan)
);

INSERT INTO Transaksi (ID_Transaksi, ID_Produk, ID_Supplier, ID_Karyawan, Jumlah, Total_Harga, Tanggal_Transaksi) VALUES
        (1, 1, 1, 1, 3, 180000, '2024-05-03 09:00:00'),
        (2, 2, 2, 2, 4, 190000, '2024-05-04 09:15:00'),
        (3, 3, 3, 3, 30, 10500000, '2024-05-05 09:30:00'),
```

```
(4, 4, 4, 4, 60, 4800000, '2024-05-06 09:45:00'),
(5, 5, 5, 5, 45, 13500000, '2024-05-07 10:00:00'),
(6, 6, 6, 6, 55, 6600000, '2024-05-08 10:15:00'),
(7, 7, 7, 7, 35, 6300000, '2024-05-09 10:30:00'),
(8, 8, 8, 8, 70, 14000000, '2024-05-10 10:45:00'),
(9, 9, 9, 9, 65, 11700000, '2024-05-11 11:00:00'),
(10, 10, 10, 10, 25, 10000000, '2024-05-12 11:15:00');
```

b) Penjelasan

Kode di atas digunakan untuk membuat tabel transaksi dan mengisi data sebanyak 10 data. Untuk membuat tabel transaksi menggunakan perintah CREATE TABLE Transaksi (namaKolom1, namaKolom2, dst.). Tabel di atas memiliki kolom yang terdiri dari id transaksi sebagai primary key, id produk yang terhubung dengan tabel produk menggunakan perintah FOREIGN KEY, id supplier yang terhubung dengan tabel supplier menggunakan perintah FOREIGN KEY, id karyawan yang terhubung dengan tabel karyawan menggunakan perintah FOREIGN KEY, jumlah, total harga, dan tanggal transaksi. Selanjutnya untuk mengisi data pada setiap kolom menggunakan perintah INSERT INTO.

4.2.2 Stored Procedure Menghtiung Total Harga Berdasarkan Karyawan

a) Source Code

```
-- Nomor 1
DELIMITER //
CREATE PROCEDURE TotalTransaksi (
   IN Karyawan INT,
   OUT TotalHarga DOUBLE
)
BEGIN
   SELECT SUM(Total_Harga) INTO TotalHarga
   FROM Transaksi
   WHERE ID_Karyawan = Karyawan;
END//
```

```
DELIMITER ;

CALL TotalTransaksi(2, @TotalHarga);
SELECT @TotalHarga;
```

b) Penjelasan

Kode di atas digunakan untuk membuat sebuah prosedur yang menggunakan parameter IN dan OUT. Kode di atas akan menampilkan jumlah total harga keseluruhan berdasarkan id karyawan menggunakan fungsi agregasi (SUM). Untuk membuat prosedur menggunakan perintah CREATE PROCEDURE nama_prosedur. IN menjadi parameter pemanggil dimana data akan dicari berdasarkan id karyawan dan OUT menjadi parameter untuk menyimpan nilai dari fungsi agregasi sehingga kode eksekusi akan dimulai dari perintah SELECT SUM(Total_Harga) INTO nama parameter OUT FROM Transaksi dimana ID_Karyawan = nama parameter IN. Untuk menampilkan datanya menggunakan syntax CALL nama_prosedur (id_karyawan yang dicari, @nama parameter OUT. @nama parameter adalah sebuah variabel yang akan diisi dengan nilai yang dikirimkan saat prosedur dipanggil.

4.2.3 Stored Procedure Lama Setiap Produk di Gudang Sejak Tanggal Masuk

a) Source Code

```
 -- Nomor 2
DELIMITER //
CREATE PROCEDURE LamaProduk()
BEGIN
   SELECT P.nama_produk, G.nama, DATEDIFF(CURDATE(),
S.tanggal_update) AS Lama_Di_Gudang
   FROM Produk P
   JOIN Stok S ON P.ID_Produk = S.ID_Produk
   JOIN Gudang G ON S.ID_Gudang = G.ID_Gudang;
END //
DELIMITER ;
```

```
CALL LamaProduk();
```

b) Penjelasan

Kode di atas digunakan untuk menampilkan data lama dalam hari setiap produk berada di gudang selama tanggal masuk. Untuk membuat prosedur menggunakan perintah CREATE PROCEDURE nama_prosedur. Prosedur ini tidak memiliki parameter, sehingga akan menjadi sebuah fungsi saja tanpa menggunakan parameter. Kode akan dimulai dari perintah untuk menampilkan nama produk, nama gudang, dan kode perhitungan selisih dalam jumlah hari (DATEDIFF) antara tanggal saat ini (CURDATE()) dan tanggal terakhir update dari tabel stok. Kode untuk menampilkan seluruh komponen tersebut berasal dari tabel stok dan gudang menggunakan perintah JOIN. Untuk menampilkan datanya dapat menggunakan syntax CALL nama_prosedur.

4.2.4 Stored Procedure Menghapus Transaksi yang Kurang Dari 200 Ribu Pada Satu Bulan Terakhir

a) Source Code

```
-- Nomor 3
DELIMITER //
CREATE PROCEDURE Hapus_Transaksi()
BEGIN
   DELETE FROM Transaksi
   WHERE Tanggal_Transaksi >= DATE_SUB(CURDATE(), INTERVAL 1
MONTH)
   AND Total_Harga <= 200000;
END //
DELIMITER ;

CALL Hapus_Transaksi();
```

b) Penjelasan

Kode di atas digunakan untuk menghapus data pada tabel transaksi yang memiliki total harga kurang dari 200000 dalam sebulan terakhir. Untuk membuat prosedur menggunakan perintah CREATE

PROCEDURE nama_prosedur. Prosedur ini tidak memiliki parameter, sehingga akan menjadi sebuah fungsi saja tanpa menggunakan parameter. Kode akan dimulai dari perintah DELETE dari tabel Transaksi dimana (WHERE) tanggal transaksinya yang dilakukan dalam satu bulan terakhir akan dihapus menggunakan fungsi DATE_SUB dan total harga diatur kurang dari 200000. Untuk menjalankan prosedur dapat menggunakan syntax CALL nama_prosedur.

4.2 Hasil

4.2.1 Data Setiap Tabel

- **Tabel Produk**

ID_Produk	Nama_Produk	Kategori_Produk	Harga	Berat
1	Kemeja Putih Polos	Kemeja	60000	0.3
2	Celana Jeans Slim Fit	Celana	47500	0.5
3	Gaun Floral Maxi	Gaun	350000	0.7
4	Kaos Oblong Basic	Kaos	80000	0.2
5	Jaket Denim Washed	Jaket	300000	0.6
6	Rok Midi A-Line	Rok	120000	0.4
7	Hoodie Sweater	Sweater	180000	0.5
8	Kemeja Tartan Flannel	Kemeja	200000	0.4
9	Celana Panjang Chino	Celana	180000	0.4
10	Blazer Formal	Blazer	400000	0.8

- **Tabel Supplier**

ID_Supplier	Nama_Supplier	Alamat	Telepon	Email
1	PT. Amanah Textile	Jl. Pahlawan No. 123, Jakarta	021-12345678	info@amanah-textile.com
2	CV. Maju Jaya Garment	Jl. Raya Industri No. 45, Bandung	022-98765432	info@maju-jaya-garment.com
3	UD. Bersama Fashion	Jl. Merdeka No. 87, Surabaya	031-56789012	info@bersama-fashion.co.id
4	PT. Sentosa Apparel	Jl. Pelangi No. 55, Semarang	024-34567890	info@sentosa-apparel.com
5	CV. Makmur Abadi Textile	Jl. Jaya No. 10, Solo	0271-2345678	info@makmur-abadi-textile.co.id
6	UD. Sejahtera Garment	Jl. Damai No. 20, Medan	061-7890123	info@sejahtera-garment.com
7	PT. Lancar Jaya Fashion	Jl. Harmoni No. 30, Malang	0341-8901234	info@lancar-jaya-fashion.com
8	CV. Bahagia Textile	Jl. Bahagia No. 5, Makassar	0411-5678901	info@bahagia-textile.co.id
9	PT. Jaya Bersama Apparel	Jl. Bersama No. 3, Palembang	0711-2345678	info@jaya-bersama-apparel.com
10	UD. Sukses Fashion	Jl. Sukses No. 8, Yogyakarta	0274-7890123	info@sukses-fashion.co.id

- **Tabel Gudang**

ID_Gudang	Nama	Alamat
1	Gudang Utama	Jl. Gatot Subroto No. 123, Jakarta
2	Gudang Pusat	Jl. Sudirman No. 45, Bandung
3	Gudang Sentral	Jl. Diponegoro No. 87, Surabaya
4	Gudang Logistik	Jl. Gajah Mada No. 55, Semarang
5	Gudang Distribusi	Jl. Pemuda No. 10, Solo
6	Gudang Amanah	Jl. Merdeka No. 20, Medan
7	Gudang Maju	Jl. Dipa No. 30, Malang
8	Gudang Bersama	Jl. Hasanuddin No. 5, Makassar
9	Gudang Sejahtera	Jl. Kapten No. 3, Palembang
10	Gudang Lancar	Jl. Pahlawan No. 8, Yogyakarta

- **Tabel Stok**

ID_Stok	ID_Produk	ID_Gudang	Jumlah	Tanggal_Update
1	1	1	100	2024-05-01 09:00:00
2	2	2	80	2024-05-02 09:15:00
3	3	3	50	2024-05-03 09:30:00
4	4	4	120	2024-05-04 09:45:00
5	5	5	70	2024-05-05 10:00:00
6	6	6	90	2024-05-06 10:15:00
7	7	7	60	2024-05-07 10:30:00
8	8	8	110	2024-05-08 10:45:00
9	9	9	85	2024-05-09 11:00:00
10	10	10	40	2024-05-10 11:15:00
NULL	NULL	NULL	NULL	NULL

- **Tabel Karyawan**

ID_Karyawan	ID_Gudang	NAMA	Alamat	Posisi	Gaji
1	1	Budi Santoso	Jl. Merdeka No. 1, Jakarta	Manajer Gudang	8000000
2	2	Ani Wijaya	Jl. Sudirman No. 2, Bandung	Staf Gudang	5000000
3	3	Cahyo Nugroho	Jl. Diponegoro No. 3, Surabaya	Staf Gudang	5000000
4	4	Dewi Kurniawan	Jl. Gajah Mada No. 4, Semarang	Staf Gudang	5000000
5	5	Eka Setiawan	Jl. Pemuda No. 5, Solo	Staf Gudang	5000000
6	6	Fita Dewanti	Jl. Merdeka No. 6, Medan	Staf Gudang	5000000
7	7	Galih Susanto	Jl. Dipa No. 7, Malang	Staf Gudang	5000000
8	8	Hani Maulana	Jl. Hasanuddin No. 8, Makassar	Staf Gudang	5000000
9	9	Indra Wibowo	Jl. Kapten No. 9, Palembang	Staf Gudang	5000000
10	10	Joko Prasetyo	Jl. Pahlawan No. 10, Yogyakarta	Staf Gudang	5000000

- **Tabel Transaksi**

ID_Transaksi	ID_Produk	ID_Supplier	ID_Karyawan	Jumlah	Total_Harga	Tanggal_Transaksi
1	1	1	1	3	180000	2024-05-03 09:00:00
2	2	2	2	4	190000	2024-05-04 09:15:00
3	3	3	3	30	10500000	2024-05-05 09:30:00
4	4	4	4	60	4800000	2024-05-06 09:45:00
5	5	5	5	45	13500000	2024-05-07 10:00:00
6	6	6	6	55	6600000	2024-05-08 10:15:00
7	7	7	7	35	6300000	2024-05-09 10:30:00
8	8	8	8	70	14000000	2024-05-10 10:45:00
9	9	9	9	65	11700000	2024-05-11 11:00:00
10	10	10	10	25	10000000	2024-05-12 11:15:00

4.2.2 Stored Procedure Menghtiung Total Harga Berdasarkan Karyawan

@TotalHarga
190000

4.2.3 Stored Procedure Lama Setiap Produk di Gudang Sejak Tanggal Masuk

nama_produk	nama	Lama_Di_Gudang
Kemeja Putih Polos	Gudang Utama	12
Celana Jeans Slim Fit	Gudang Pusat	11
Gaun Floral Maxi	Gudang Sentral	10
Kaos Oblong Basic	Gudang Logistik	9
Jaket Denim Washed	Gudang Distribusi	8
Rok Midi A-Line	Gudang Amanah	7
Hoodie Sweater	Gudang Maju	6
Kemeja Tartan Flannel	Gudang Bersama	5
Celana Panjang Chino	Gudang Sejahtera	4
Blazer Formal	Gudang Lancar	3

4.2.4 Stored Procedure Menghapus Transaksi yang Kurang Dari 200 Ribu Pada Satu Bulan Terakhir

ID_Transaksi	ID_Produk	ID_Supplier	ID_Karyawan	Jumlah	Total_Harga	Tanggal_Transaksi
3	3	3	3	30	10500000	2024-05-05 09:30:00
4	4	4	4	60	4800000	2024-05-06 09:45:00
5	5	5	5	45	13500000	2024-05-07 10:00:00
6	6	6	6	55	6600000	2024-05-08 10:15:00
7	7	7	7	35	6300000	2024-05-09 10:30:00
8	8	8	8	70	14000000	2024-05-10 10:45:00
9	9	9	9	65	11700000	2024-05-11 11:00:00
10	10	10	10	25	10000000	2024-05-12 11:15:00
NULL	NULL	NULL	NULL	NULL	NULL	NULL

BAB V
PENUTUP

5.1 Analisa

Dari hasil praktikum, praktikan menganalisa bahwa penggunaan tipe data yang tepat dalam sistem manajemen basis data (DBMS) adalah kunci untuk memastikan keakuratan dan konsistensi data. Dengan memilih tipe data yang sesuai, seperti string untuk teks, integer untuk angka bulat, atau float untuk angka desimal, DBMS dapat mengelola data dengan efisien dan mengoptimalkan penggunaan penyimpanan. Misalnya, menggunakan tipe data yang lebih kecil untuk kolom-kolom yang membutuhkan kapasitas penyimpanan yang lebih kecil dapat menghemat ruang penyimpanan dan meningkatkan kinerja sistem.

Penerapan stored procedure dalam sistem manajemen basis data memberikan sejumlah manfaat yang signifikan. Stored procedure memungkinkan pengguna untuk mengelompokkan serangkaian perintah SQL menjadi satu unit logis, yang dapat dipanggil dan dieksekusi secara berulang tanpa perlu menulis ulang kode. Selain itu, stored procedure juga membantu meningkatkan keamanan data dengan mengizinkan kontrol akses yang ketat. Dengan menyimpan logika bisnis di dalam stored procedure, pengguna dapat mengontrol akses langsung ke tabel-tabel basis data, sehingga mengurangi risiko eksploitasi dan kebocoran data yang tidak diinginkan.

5.2 Kesimpulan

1. Tipe data merupakan bagian dari variabel yang mempengaruhi perilaku variabel. Dengan tipe data ini, bisa ditentukan nilai apa yang bisa disimpan didalamvariabel tersebut.
2. Stored Procedure adalah sebuah prosedur layaknya subprogram (subrutin) di dalam bahasa pemrograman reguler yang tersimpan di dalam katalog basis data.
3. Dengan demikian, penggunaan tipe data yang tepat dan stored procedure dalam basis data tidak hanya meningkatkan efisiensi operasional, tetapi juga meningkatkan keamanan dan keandalan sistem secara keseluruhan.