Penggunaan Fungsi *(Function)* Pada Rancangan BasisData Aplikasi Penjualan

Sigit Prasetyo Karisma Utomo[1)], Jimi Asmara[2)], Praditya Kurniawan[3)]

[1), 2),3)] Magister Teknik Informatika STMIK AMIKOM Yogyakarta
Jl Ring road Utara, Condongcatur, Sleman, Yogyakarta 55281
Email : sigitprasetyokarismautomo@gmail.com[1)], jimyasmara@ymail.com[2)], pradityakurniawan@gmail.com [3)]

Abstrak

Sebuah sistem basisdata pada dasarnya adalah komputerisasi sistem penyimpanan catatan. Basidata itu sendiri dapat dianggap sebagai jenis lemari arsip elektronik.pengguna sistem ini dapat melakukan beragam operasi terdahap berkas. Penggunaan fungsi (*function*) pada basisdata dapat membuat *constraint check* yang berfungsi sebagai filter data sebelum dimasukkan ke dalam tabel sehingga data dalam tabel valid.

Kata kunci: basisdata,fungsi, constraint chek.

1. Pendahuluan

1.1.Latar Belakang

Basisdata merupakan sebuah komputerisasi sistem penyimpanan *record*, sebuah sistem komputerisasi yang tujuan keseluruhannya menyimpan informasi dan menginjinkan untuk mengambil kembali dan memperbarui informasi tersebut atas permintaan pengguna. Pada saaat sekarang sudah menggunakan Relational Database Manajement Sytem (RDBMS), banyak aplikasi database seperti MySQL Server, MySQL, PostgreSQL,Oracle Sybase dan sebagainya.
Sistem yang baik harus memiliki basisdata yang baik sehingga dalam mengolah data-data tersebut tidak terjadi redudansi data. Basisdata yang baik juga diharuskan untuk memenuhi kebutuhan informasi dari pengguna basidata tersebut.
Aplikasi penjualan adalah suatu bentuk pengolahan data untuk membantuk mempercepat transaksi antara penjual dan pembeli dan mencatatnya didokumen-dokumen dasar memasukannya kedalam sistem informasi dan merekamnya kedalam basisdata dan mengolahnya menjadi informasi-informasi pencatatan nilai dan mengahasilkan laporan.

Basisdata adalah sebuah koleksi dari data yang tahan lama yang digunakan oleh sistem aplikasi dari perusahaan tertentu.dengan tahan lama data basisdata berbeda jenis dengan data yang berlansung singkat, seperti data input,data output,kunci kontrol,antrian pekerjaan,blok kontrol pernagkta lunak ,hasil menengah dan lebih umum data apa saja yang sifatnya sementara.[1]

Basis data juga dituntuk untuk memenuhi kebutuhan fungsi dan informasi dari penggunannya,sistem yang baik adalah sistem yang dapat mengolah data dengan baik sehingga dapat mempermudah pekerjaan user.

Fungsi (*function*) merupakan fasilitas yang disediakan oleh database secara umum untuk memanipulasi data yang kompleks, seperti mencari jumlah data, mencari rata-rata nilai yang ada dalam suatu tabel dan lain-lain, bukan hanya sekedar membaca data dari table saja. Fungsi adalah suatu rutin khusus yang disediakan oleh MySQL untuk melakukan manipulasi suatu data[2]

Pada dasarnya sebuah database terdiri dari satu tabel atau lebih. Tabel sendiri tersususn dari satu record atau lebih.*Record* terdiri dari satu atau lebih kolom.tiap kolom bisa berisi data yang berbeda-beda tipenya[3].

1.2.Tinjauan pustaka.

Basis kurang lebih diartikan sebagai markas/gudang, tempat bersarang/berkumpul, sedangkan Kata data diambil dari bahasa latin untuk"memberi"; jadi data sesungguhny adalah fakta yang diberikan,darimana kenyataan tambahan dapat ditarik kesimpulannya[1].

Kita dapat mencoba mengelola data dengan menyimpan dalam file sistem operasi .pendekatan ini mempunyai banyak kelemahan meliputi:

1. Kita mungkin tidak memilki meori utama sebesar 500GB untuk menyimpa semua data,oleh karena itu kita harus menyimpan data dalam alat penyimpan data seperti *harddisk*[4].
2. Sekalipun kita mempunyai memori utama 500 GB,pada sistem komputer dengan 32*bit addresing* kita tidak dapat menunjuk langsung pada data yang lebih dari 4GB[4].
3. Kita harus menulis program khusus untuk menjawab setiap pertanyaan yang ingin pengguna tanyakan mengenai data[4].
4. Kita harus melindungi data dari perubahan tidak konsisten yang dilakukan oleh penggunayang berbeda yang mengakses data secara konkuren[4].
5. Kita harus memastikan data di-*restore* dalam keaadaan yang konsisten jika sistem mengalami *crash* saat perubahan sedang dilakukan[4].
6. Sistem operasi hanya menyediakan mekanisme *password* untuk keamanan[4].

Adapun manfaat dari basisdata adalah sebagai berikut:

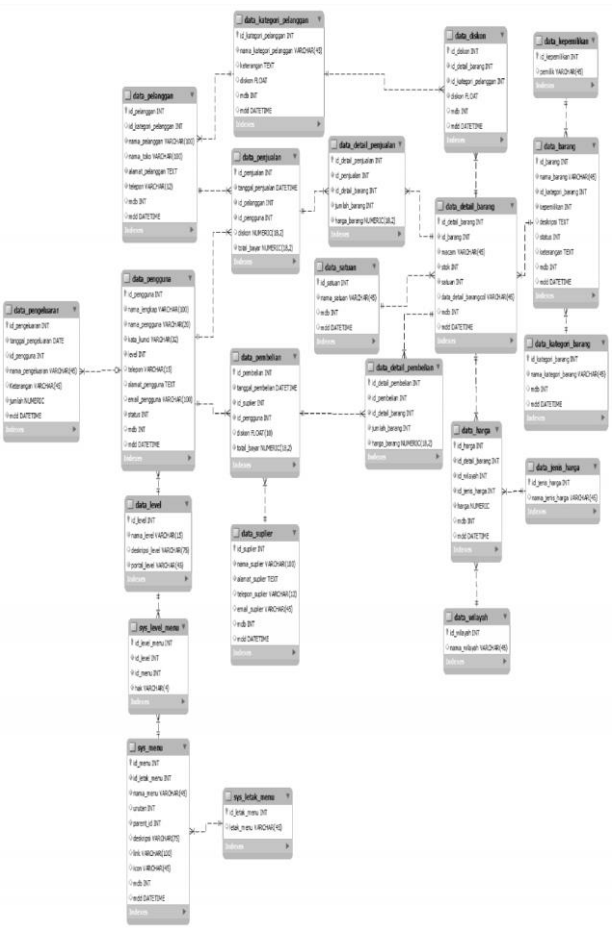
1. Saling berbagi data.
   Berbagi tidak hanya berarti bahwa aplikasi yang telah ada dapat memakai bersama data dalam basisdata, tetapi juga bahwa aplikasi baru dapat dikembangkan untuk beroperasi terhadap data yang sama.[1]
2. Redudansi dapat dikurangi..
   Redudansi terjadi jika suatu informasi disimpan di beberapa tempat. Apabila terjadi suatu redudansi makan solusi yang dianggap tepat yaitu Normalisasi.[1]
3. Sifat tidak konsisten bisa dihindari.
   Sifat tidak konsisten antara dua masukan yang pokok isinya untuk menggambar fakta.[1]
4. Dukungan transaksi dapat disedikan.[1]
   Transaksi merupakan sebuah unit kerja logis secara tipikal melibatkan beberapa operasi basisdata.
5. Integritas dapat dipertahankan[1].
   Masalah integritas adalah masalah memastikan bahwa data dalam basisdata adalah benar[1].
6. Keamanan dapat dijalankan
   Keamanan atau aturan untuk diperiksa saat percobaan akses pada data sensitive[1].
7. Persyaratan yang bertentangan bisa diseimbangkan[1].
8. Standar bisa dijalankan.
   Membuat standar gambaran data khususnya disukai sebagai sebuah bantuan pada pertukaran data[1].

Pengelompokkan fungsi *(function)* didasarkan pada kegunaan dari fungsi-fungsi tersebut antara lain:

1. Fungsi Sistem
   Kelompok fungsi sistem adalah kelompok yang memberikan informasi tentang pemakaian server database oleh pemakai[2].
2. Fungsi Agregat
   Fungsi agregat adalah fungsi standar di dalam SQL, suatu fungsi yang digunakan untuk melakukan summary, merupakan fungsi stastistik standar yang dikenakan pada suatu tabel atau query[2].
3. Fungsi Aritmetika
   MySQL memiliki fasilitas dasar untuk melakukan manipulasi data numerik, seperti penjumlahan, pengurangan, perkalian dan pembagian yang disertakan dalam suatu perintah select. Fasilitas tersebut dikenal dengan operator aritmatika[2.]
4. Fungsi Tanggal
   Tanggal dalam MySQL menggunakan tanggal dari sistem Unix, tidak ada masalah sampai dengan 2069. Semua tahun yang dinyatakan dengan dua digit diasumsikan tahun tersebut pada range tanggal antara tahun 1970 sampai dengan 2069. Jika dimasukkan tahun 01 maka akan dianggap tahun 2001. Format tanggal dan jam dalam MySQL adalah dengan menggunakan format tahun, bulan, tangal, jam, menit dan detik[2].
5. Fungsi *String*
   Fungsi *strin*g digunakan untuk manipulasi teks *(string).* MySQL menyediakan banyak fungsi built-in untuk melakukan manipulasi teks ini[2]
6. Fungsi Logika
   Fungsi logika merupakan fungsi yang disediakan oleh MySQL untuk melakukan evaluasisuatu ekspresi. Berdasarkan nilai ekspresi ini akan dihasilkan suatu nilai yang akan ditampilkan pada hasil *query*[2].

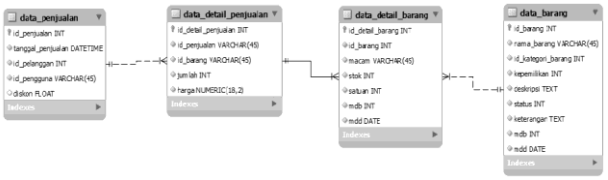
2. Hasil dan Pembahasan.

Kasus yang diambil adalah rancangan basisdata pada aplikasi penjualan barang berbasis web. Secara keseluruhan rancangan basisdata pada aplikasi penjualan dapat dilihat pada Gambar 1.



Gambar 1. *Rancangan Basis Data pada Aplikasi Penjualan.*

Agar pembahasan yang dilakukan lebih rinci, penulis hanya fokus pada bagian penjualan barang. Rancangan tabel yang digunakan dalam pembuatan *function* ini adalah tabel data barang, tabel detail data barang, tabel detail penjualan, dan tabel penjualan. Sehingga secara sederhana relasi yang terbentuk menggunakan 4 tabel tersebut dapat dilihat pada Gambar 2.



Gambar 2. *Rancangan Tabel Sederhana yang Akan Digunakan Untuk Membahas Function*

2.1. Pembuatan Database.

Setelah mengetahui gambaran rancangan tabel yang akan digunakan, penulis melakukan pembuatan basisdata menggunakan DBMS *PostgreSQL*. Berikut langkah – langkah yang penulis gunakan untuk membuat basisdata.

-Membuat database

```
CREATE DATABASE penjualan;
```

- Membuat tabel Data Barang

```
CREATE TABLE IF NOT EXISTS
 data_barang (
 id_barang SERIAL PRIMARY KEY,
 nama_barang VARCHAR(45) NOT NULL,
 id_kategori_barang INT NOT NULL ,
 kepemilikan INT NOT NULL ,
 deskripsi TEXT NOT NULL ,
 status INT NOT NULL ,
 keterangan TEXT NOT NULL ,
 mdb INT NOT NULL ,
 mdd DATE NOT NULL);
```

- Membuat tabel Data Detail Barang

```
-CREATE TABLE IF NOT EXISTS
 data_detail_barang (
 id_detail_barang SERIAL PRIMARY KEY,
 id_barang INT NOT NULL ,
 macam VARCHAR(45) NOT NULL ,
 stok INT NOT NULL ,
 satuan INT NOT NULL ,
 mdb INT NOT NULL , mdd
 DATE NOT NULL , FOREIGN
 KEY (id_barang)
 REFERENCES data_barang (id_barang)
 ON DELETE NO ACTION
 ON UPDATE NO ACTION)
```

- Membuat tabel Data Penjualan

```
-CREATE TABLE IF NOT EXISTS
 data_penjualan (
 id_penjualan SERIAL PRIMARY KEY,
 tanggal_penjualan TIMESTAMP NOT NULL
 ,
 id_pelanggan INT NOT NULL ,
 id_pengguna INT NOT NULL ,
 diskon FLOAT NULL)
```

- Membuat tabel Data Detail Penjualan

```
-CREATE TABLE IF NOT EXISTS
 data_detail_penjualan (
 id_detail_penjualan SERIAL PRIMARY
 KEY ,
 id_penjualan INT NOT NULL ,
 id_barang INT NOT NULL ,
 jumlah INT NOT NULL ,
 harga DECIMAL(18,2) NOT NULL ,
 FOREIGN KEY (id_penjualan)
 REFERENCES data_penjualan
 (id_penjualan)
 ON DELETE NO ACTION ON
 UPDATE NO ACTION,
 FOREIGN KEY (id_barang)
 REFERENCES data_detail_barang
 (id_detail_barang)
 ON DELETE NO ACTION
 ON UPDATE NO ACTION)
```

- Menambahkan dummy data

id_barang integer	nama_barang character varying(45)	id_kategori_barang integer	kepemilikan integer	deskripsi text	status integer	keterangan text	mdb integer	mdd date
1	Sabun Mandi	1	1	Sabun untuk Mandi	1	Milik Sendiri	1	2014-12-09
2	Sabun Cuci	1	1	Sabun untuk Cuci	1	Milik Sendiri	1	2014-12-09
3	Sabun Muka	1	1	Sabun untuk Muka	1	Milik Sendiri	1	2014-12-09
4	Sampo	1	1	Sampo	1	Milik Sendiri	1	2014-12-09

Gambar 3. *Tabel barang.*

id_detail_barang integer	id_barang integer	macam character varying(45)	stok integer	satuan integer	mdb integer	mdd date
1	1	Aroma Buah	100	1	1	2014-12-09
2	1	Aroma Bunga	120	1	1	2014-12-09
3	2	Cuci Tangan	130	1	1	2014-12-09
4	2	Mesin Cuci	100	1	1	2014-12-09
5	2	Cuci Extra	75	1	1	2014-12-09
6	3	Muka Berminyak	100	1	1	2014-12-09
7	3	Muka Kering	100	1	1	2014-12-09
8	4	Aroma Apel	50	1	1	2014-12-09
9	4	Segar	80	1	1	2014-12-09
10	4	Mint	100	1	1	2014-12-09

Gambar 4. *Tabel detail barang.*

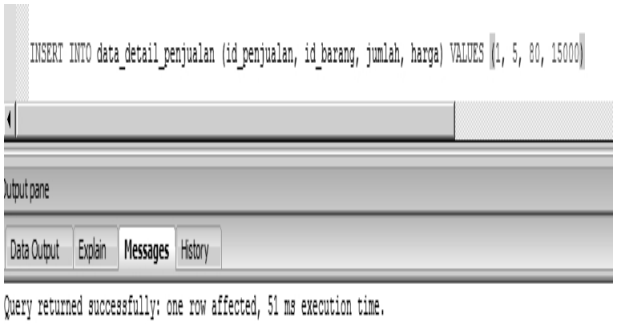
Pada alur penjualan, barang yang dibeli pada saat proses transaksi akan tersimpan pada tabel detail penjualan. Secara logika pembeli tidak dapat membeli sebuah produk melebihi batas stok yang ada. Jika database belum menggunakan *constraint check* maka pembelian sebuah barang dapat melebihi stok yang ada. Penulis simulasikan sebagai berikut. Seorang *user* melakukan sebuah transaksi pembelian. *Query* yang berjalan saat pembuatan transaksi adalah sebagai berikut:

```
INSERT INTO data_penjualan
  (tanggal_penjualan, id_pelanggan,
  id_pengguna) VALUES (NOW(), 1, 1)
```

penulis melakukan pengujian dengan melakukan pembelian diatas stok yang tersedia. Sebagai contoh dalam transaksi sebelumnya melakukan pembelian barang sabun cuci dengan macam cuci extra sebanyak 80 (stok yang tersedia 75). Query yang digunakan adalah sebagai berikut:

```
INSERT INTO data_detail_penjualan
  (id_penjualan, id_barang, jumlah,
  harga) VALUES (1, 5, 80, 15000)
```

Query tersebut masih dapat dieksekusi dengan baik oleh DBMS. Seharusnya *query* tersebut tidak dapat dieksekusi karena jumlah pembelian melebihi stok. Hasil eksekusi *query* dapat dilihat pada gambar 5.



Gambar 5. *Hasil Eksekusi Query*

3. Implementasi

Dari permasalahan diatas dapat dibuat sebuah *constraint check* menggunakan *function* untuk mengecek jumlah stok pada tabel data detail barang sebelum di *insert* kan pada tabel detail penjualan. *Function* yang dibuat adalah sebagai berikut:

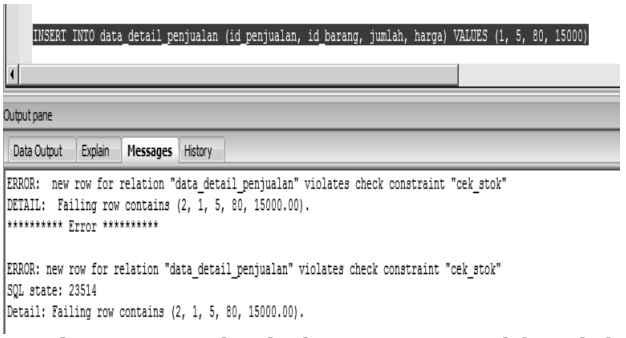
```
CREATE OR REPLACE FUNCTION check_stock
  ( id int, jumlah int )
RETURNS boolean
AS $$
DECLARE
sisa INT;
BEGIN
  SELECT (stok - jumlah) into sisa
  FROM data_detail_barang WHERE
  id_detail_barang = id;
IF(sisa >= 0)
  THEN RETURN true;
ELSE
  RETURN false;
END IF;
END;
$$
LANGUAGE plpgsql;
```

Setelah itu lakukan perubahan struktur tabel data_detail_penjualan untuk ditambahkan *constraint check* menggunakan function yang baru dibuat. Sebelum melakukan penambahan *constraint check* harus melakukan pengosongan data pada tabel dapat menggunakan perintah *truncate*.

truncate table data_detail_penjualan;
Lakukan perubahan struktur tabel dan ditambahkan *constraint check.*

```
ALTER TABLE data_detail_penjualan
ADD CONSTRAINT cek_stok CHECK
(check_stock(id_barang, jumlah))
```

Setelah melakukan penambahan *constraint check* pada tabel data_detail_penjualan, lalu jalankan skenario permasalahan awal tadi. Pada saat *Query* dijalankan menghasilkan *error* sehingga data tidak di *insert* kedalam tabel. Hasil eksekusi *query* setelah tabel ditambahkan *constraint check* dapat dilihat pada gambar 6.



Gambar 6. Hasil eksekusi *query* setelah tabel ditambahkan *constraint check*

4. Kesimpulan.

a. Dari kasus diatas dapat diambil kesimpulan bahwa penggunaan *function* dalam merancang basisdata dapat digunakan untuk menyaring *(filter)* data yang akan dimasukkan kedalam tabel sehingga data dalam tabel *valid*. Selain itu, function dapat dikembangkan lebih jauh dengan menggunakan *trigger*, sehingga saat sebuah barang dibeli dengan jumlah tertentu yang masih dalam jumlah stok akan otomatis mengurangi jumlah stok yang tersedia.

b. Kasus diatas hanyalah sebuah contoh yang digunakan untuk memahami penggunaan *function* pada perancangan basis data. Pada kasus tersebut masih banyak yang dapat dikembangkan agar rancangan basisdata yang digunakan dapat memvalidasi data sebelum di masukkan pada suatu tabel.

Biodata Penulis.

Sigit Prasetyo Karisma Utomo, memperoleh gelar Sarjana Komputer (S.Kom), Jurusan Teknik Informatika STMIK AMIKOM Yogyakarta, lulus tahun 2014. Saat ini sedang menempuh Program Pasca Sarjana Program Studi Magister Teknik Informatika STMIK AMIKOM Yogyakarta

Jimi Asmara, memperoleh gelar Sarjana Komputer (S.Kom), Jurusan Sistem Informasi STIKOM Uyelindo Kupang, lulus tahun 2012 Saat ini sedang menempuh Program Pasca Sarjana Program Studi Magister Teknik Informatika STMIK AMIKOM Yogyakarta

Praditya Kurniawan, memperoleh gelar Sarjana Komputer (S.Kom), Jurusan Teknik Informatika STMIK AMIKOM Yogyakarta, lulus tahun 2014. Saat ini sedang menempuh Program Pasca Sarjana Program Studi Magister Teknik Informatika STMIK AMIKOM Yogyakarta

Daftar Pustaka

[1] C.J. Date ”pengenalan basisdata,” edisi ke tujuh jilid 1, Indeks Group Gramedia,2004.
[2] Libre,Fitria”Modul Basisdata,” D3 FMIPA UGM.
[3] Suja, Iman ”Pemograman SQL dan Database Server MySQL,”Yogyakarta:Andi Offset,2005.
[4] Ramakrisnan Raghu dan Johanes Gehrke,”*Sistem Manajemen Basisdata*, “Yogyakarta:Andi Offset,2004.