

LAPORAN

ANALISIS DAN PERANCANGAN SISTEM INFORMASI PENJUALAN PADA MINIMARKET 212 BERBASIS SQL

Agung Dwi Cahyono¹, Firman Yudha Prawira², Shavira Adianda Octobiana³
Program Studi Jaringan Telekomunikasi Digital
fyudhaprawira@gmail.com
sesekaranofficial@gmail.com
Shaviraadianda123@gmail.com

ABSTRAK

Minimarket 212 merupakan salah satu minimarket yang menjual berbagai macam kebutuhan rumah tangga. Minimarket dalam melakukan pencatatan transaksi penjualan masih menggunakan cara manual dengan menuliskan di buku pencatatan transaksi penjualan. Pencatatan yang dilakukan ini memungkinkan terjadinya kesalahan maupun kelalaian. Berdasarkan permasalahan tersebut ini akan membuat sebuah sistem informasi yang mencatat penjualan pada Minimarket. Tujuannya untuk memudahkan proses informasi penjualan pada Minimarket 212.

Sistem informasi penjualan merupakan metode yang dirancang untuk menghasilkan, menganalisa, mengedarkan, dan memperoleh informasi guna mendukung pengambilan keputusan mengenai penjualan. Tahap pengembangan aplikasi meliputi analisis, perancangan sistem, implementasi dan pengujian. Rancangan tersebut telah diimplementasikan menjadi Sistem Informasi Penjualan dengan bahasa pemrograman PHP dan database MySQL.

Kata Kunci : SQL, PHP

1. PENDAHULUAN

Minimarket adalah semacam toko kelontong yang menjual segala macam barang dan makanan, namun tidak selengkap dan sebesar sebuah supermarket. Minimarket banyak diminati oleh masyarakat karena lengkapnya produk yang dijual, kenyamanan berbelanja dan kebebasan memilih produk sendiri tanpa perlu banyak dilayani. Semakin lengkap variasi produk yang ditawarkan kepada konsumen maka akan semakin banyak pula pilihan barang yang akan dibeli oleh konsumen sesuai dengan kebutuhannya. Salah satu Minimarket yang sedang ditunjukkan ialah .Minimarket 212 berlokasi di Malang Raya. Produk yang ditawarkan antara lain peralatan rumah tangga, kebutuhan rumah tangga ditambah dengan produk obat, baju, alat elektronik dan olahan makanan lainnya.

Minimarket 212 berusaha memenuhi kebutuhan konsumen dengan memperlengkap jenis barang yang dijual. Namun dalam menjalankan usahanya, pihak Minimarket masih juga menggunakan cara manual dengan dalam pencatatan transaksi penjualan. Pihak Minimarket harus mendata apa saja yang dibeli oleh konsumen di buku pencatatan transaksi penjualan. Sehingga konsumen harus menunggu agak lama ketika melakukan pembayaran di kasir. Selain itu data stok barang yang ada hanya disimpan ke dalam Microsoft Excel, belum disimpan ke database. Mereka harus mengupdate data secara manual ketika barang atau produk tersebut laku terjual. Kemudian dalam pencatatan Minimarket yang masih menggunakan ms.excel untuk pendataan sehingga dianggap tidak efisien.

Pencatatan yang dilakukan ini memungkinkan terjadinya kesalahan maupun kelalaian (human error) , mengupdate data produk/barang atau mencatat transaksi penjualan ke dalam buku, hal ini juga dapat menimbulkan kesalahan pahaman dengan pemilik Minimarket. Maka dari

itu dibutuhkan suatu teknologi untuk memberi keefektifan dan juga untuk mengurangi terjadinya human error.

Dengan kemajuan teknologi sangat memungkinkan untuk membuat suatu sistem informasi penjualan Minimarket dengan memanfaatkan teknologi basis data dan website sebagai pengaplian. Teknologi ini membantu dalam membuat, membaca, mengapdet dan menghapus data dengan cepat dan akurat. Melalui sistem informasi tersebut Minimarket dapat memiliki peluang lebih berkembang dan perkembangan Minimarket dapat terpantau secara terperinci, sehingga lebih efektif..

Berdasarkan masalah tersebut diatas, dibuat dengan judul “Analisis dan Perancangan Sistem Informasi Penjualan pada Minimarket 212 Berbasis MYSQL ”. diharapkan dapat membantu Minimarket 212 dapat membuat suatu sistem yang lebih baik yang membantu dalam penginformasian penjualan minimarket 212.

2. TINJAUAN PUSTAKA

MySQL adalah program database server yang mampu menerima dan mengirimkan datanya sangat cepat, multi user serta menggunakan perintah dasar SQL (Structured query Language). MySQL merupakan dua bentuk lisensi, yaitu FreeSoftware dan Shareware. MySQL yang biasa kita gunakan adalah MySQL FreeSoftware yang berada dibawah Lisensi GNU/GPL (General Public License).MySQL Merupakan sebuah database server yang free, artinya kita bebas menggunakan database ini untuk keperluan pribadi atau usaha tanpa harus membeli atau membayar lisensinya. MySQL pertama kali dirintis oleh seorang programmer database bernama Michael Widenius . Selain database server, MySQL juga merupakan program yang dapat mengakses suatu database MySQL yang berposisi sebagai Server, yang berarti program kita berposisi sebagai Client.

Jadi MySQL adalah sebuah database yang dapat digunakan sebagai Client maupun server.

Database MySQL merupakan suatu perangkat lunak database yang berbentuk database relasional atau disebut Relational Database Management System (RDBMS) yang menggunakan suatu bahasa permintaan yang bernama SQL (Structured Query Language). SQL (Structured Query Language) adalah sebuah bahasa permintaan database yang terstruktur. Bahasa SQL ini dibuat sebagai bahasa yang dapat merelasikan beberapa tabel dalam database maupun merelasikan antar database.

SQL dibagi menjadi tiga bentuk Query, yaitu :

1. DDL (Data Definition Language)

DDL adalah sebuah metode Query SQL yang berguna untuk mendefinisikan data pada sebuah Database, Query yang dimiliki DDL adalah :

a.CREATE: Digunakan untuk membuat Database dan Tabel

b.Drop : Digunakan untuk menghapus Tabel dan Database

c.Alter : Digunakan untuk melakukan perubahan struktur tabel yang telah dibuat, baik menambah Field (Add), mengganti nama Field (Change) ataupun menamakannya kembali (Rename), dan menghapus Field (Drop).

2.DML (Data Manipulation Language)

DML adalah sebuah metode Query yang dapat digunakan apabila DDL telah terjadi, sehingga fungsi dari Query DML ini untuk melakukan manipulasi database yang telah dibuat. Query yang dimiliki DML adalah :

a.INSERT: Digunakan untuk memasukkan data pada Tabel Database

b.UPDATE: Digunakan untuk pengubahan terhadap data yang ada pada Tabel Database

c.DELETE: Digunakan untuk Penhapusan data pada tabel Database

3.DCL (Data Control Language)

DCL adalah sebuah metode Query SQL yang digunakan untuk memberikan hak otorisasi mengakses Database, mengalokasikan space, pendefinisian space, dan pengauditan penggunaan database. Query yang dimiliki DCL adalah :

a.GRANT: Untuk mengizinkan User mengakses Tabel dalam Database.

b.REVOKE: Untuk membatalkan izin hak user, yang ditetapkan oleh perintah GRANT

c.COMMIT: Menetapkan penyimpanan Database

d. ROLLBACK: Membatalkan penyimpanan Database

PHP

PHP

PHP adalah bahasa pemrograman script server-side dengan desain untuk pengembangan web. PHP dikembangkan pada tahun 1995 oleh Rasmus Lerdorf, dan sekarang dikelola oleh The PHP Group. PHP istilah bahasa pemrograman server side karena PHP diproses pada komputer server. Hal ini berbeda dibandingkan bahasa pemrograman yaitu client-side contoh JavaScript yang diproses pada web browser (client). PHP kepanjangan dari PHP Hypertext Preprocessor, dari suatu kepanjangan rekursif, yaitu kata dimana kepanjangannya dari suatu singkatan itu sendiri. PHP ini memiliki sifat Open Source. PHP dirilis di lisensi PHP License, sedikit berbeda dengan lisensi GNU General Public License (GPL) yang biasa digunakan untuk proyek Open Source.

Untuk membuat halaman web, sebenarnya PHP bukanlah bahasa pemrograman yang wajib digunakan. bisa dengan membuat website hanya menggunakan HTML saja. Web yang dihasilkan dengan HTML (dan CSS) dikenal dengan website statis yang dimana konten dan halaman web bersifat tetap.

Sebagai perbandingan, website dinamis yang bisa dibuat menggunakan PHP adalah situs web yang bisa menyesuaikan tampilan konten tergantung situasi. Website dinamis juga bisa menyimpan data ke dalam database, membuat halaman yang berubah-ubah sesuai input dari user, memproses form, dll. Untuk pembuatan web, kode ini di sisipkan ke dalam dokumen HTML. Karena fitur ini PHP dikatakan sebagai Scripting Language atau bahasa pemrograman script.

Struktur Dasar PHP

-PHP mempunyai struktur yang sederhana. Syntax PHP dimulai dengan tanda <?php dan di akhiri dengan ?>

- Saat membuat baris-baris program didalam PHP dapat menggunakan fungsi komentar untuk menjelaskan maksud dari setiap baris atau function .Komentar di PHP diawali dengan tanda //. PHP tidak akan mengeksekusi setiap karakter yang terdapat dibelakang tanda //. Karena PHP akan membaca hal tersebut merupakan komentar.

- Didalam PHP fungsi-fungsi seperti if, for, while, echo, print dan lain-lain tidak case sensitive. Penggunaan fungsi-fungsi tersebut dapat di gunakan dengan huruf besar maupun kecil. Contohnya fungsi echo pada PHP ketika membuat fungsi echo dan ECHO, PHP akan membaca kedua syntax tersebut adalah sama.

3 PERENCANAAN SISTEM DAN HASIL

Dalam melakukan suatu perancangan sistem, maka diperlukan :

1. membuat rancangan misi

-Mission statement

Mendefinisikan tujuan utama dari database yaitu merancang basis data berupa informasi, yang mendukung pengelolaan data dan juga memberi manfaat dari aplikasi database yaitu mempermudah dalam pengelolaan data dan penyajian data yang penyajian data tersebut mempermudah pengguna untuk mendapatkan data/informasi.

-Mission objective

Setelah mengidentifikasi mission statement, maka akan dibuatkan mission objective. Mission Objective berguna untuk mencantumkan semua data yang akan dikelola dalam penggunaan aplikasi sistem basis data.

Mission Objective pada sistem informasi penjualan adalah sebagai berikut :

- *Mengelola (Insert, Update, Delete) data jual*
- *Mengelola (Insert, Update, Delete) data detail jual*
- *Mengelola (Insert, Update, Delete) data barang*
- *Mengelola (Insert, Update, Delete) data user*

Mengelola (Insert, Update, Delete) data kategori

- *Menampilkan jual*
- *Menampilkan detail jual*
- *Menampilkan barang*
- *Menampilkan user*
- *Menampilkan kategori*
- *Melakukan Pencarian pada data jual*
- *Melakukan Pencarian pada data detail jual*
- *Melakukan Pencarian pada data barang*
- *Melakukan Pencarian pada data user*
- *Melakukan Pencarian pada data kategori*

2. Membuat desain rancangan.

Rancangan disusun menggunakan konteks dan diagram relasional

- Konteks

konteks berisi gambaran umum (secara garis besar) sistem yang akan dibuat. Secara kalimat, dapat dikatakan bahwa diagram konteks ini berisi "siapa saja yang memberi data (dan data apa saja) ke sistem, serta kepada siapa saja informasi (dan informasi apa saja) yang harus dihasilkan sistem."

yang dibutuhkan adalah sebagai berikut:

- 1)Siapa saja pihak yang akan memberikan data ke sistem,
- 2)Data apa saja yang diberikannya ke sistem,
- 3)kepada siapa sistem harus memberi informasi atau laporan, dan
- 4)Apa saja isi/ jenis laporan yang harus dihasilkan sistem.

Beberapa kemungkinan (data) yang diberikan pembeli kepada kasir adalah sebagai berikut.

- 1)Barang yang ditanyakan,
- 2)Barang yang akan dibeli, dan
- 3)Uang pembayaran.

Sebaliknya, kemungkinan informasi yang diberikan kasir kepada pembeli adalah sebagai berikut.

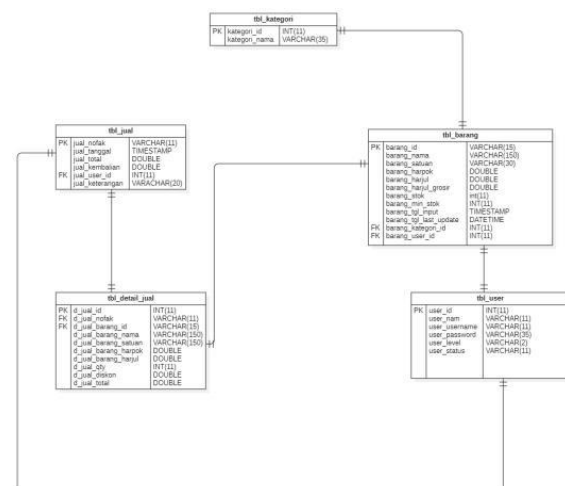
- 1)Keadaan barang yang ditanyakan,
- 2)Jumlah uang yang harus dibayar.

Sedangkan informasi yang diberikan kasir kepada Pemilik adalah Laporan Jumlah Uang Masuk beserta Jumlah Barang yang Terjualnya serta penanggung jawab.

diagram Konteks utama (transaksi barang/ jual beli)



diagram relasional ER:



Analisa Sistematisa rancangan :

Dari suatu sistem penjualan di minimarket 212, sistem dapat menampilkan informasi yang dibutuhkan didalam minimarket 212, dan memantau dengan laporan didalamnya. Perancangan menggunakan 2 diagram, dimana dengan diagram konteks dan diagram relasional ER untuk membantu adanya percangan yang berkesinambungan. Untuk perancangan ini, berawal dari

pertama, transaksi jual beli yang dilakukan antar pembeli dan penjual. Dimana pembeli membeli barang kemudian memberikan kepada pegawai lalu pegawai mendata dan melaporkan. Dengan sistem ini dipermudah untuk proses transaksi. Dimana pembeli memberikan barang yang dibeli kepada kasir/user lalu kasir/user mendatakan barang yang dibeli dengan sistem dan harga yang dibayar oleh pembeli, dengan beberapa atribut pendukung . kemudian kedua, detail jual . jadi seperti lebih ke sistem jual Produk yang merekap perjualbelian Kemudian detail ini dengan beberapa atribut yang sudah dirancang di diagram ER, yang membantu sistem penjualan barang tersebut di minimarket dengan juga membantu memberi detail penjualan pada hari itu. ketiga yaitu merancang barang. Dimana barang/produk minimarket yang merupakan hal yang dipasarkan dalam penjualan di minimarket. hal ini memberi informasi barang tersebut untuk sebagai stok barang dan harga yang didapatkan di minimarket itu sendiri. Empat, pegawai/user. Data pegawai/user dibutuhkan untuk sebagai pertanggung jawaban di dalam minimarket pada saat jam kerja. Lima, kategori. Yang berisi kategori barang termasuk barang jenis apa yang membantu dalam penge-rak-an/penjenisan barang kumpul. Hal ini berkesinambungan dengan barang.

3. membuat tabel identifikasi entitas.

| No | Nama Entitas | Deskripsi | Kegiatan |
|----|--------------|--|--|
| 1. | Jual | Merupakan entitas yang berisi jual beli /transaksi barang minimarket 212 | Merupakan penjualan/transaksi barang di minimarket 212 |
| 2. | Detail jual | Merupakan entitas yang berisi informasi detail atau lengkap meliputi sistem penjualan barang minimarket 212. | merupakan sistem informasi lengkap / detail jual barang minimarket 212 |
| 3. | barang | Merupakan entitas yang berisi informasi barang yang diperjualbelikan di minimarket 212 | Merupakan barang yang diperjual belikan dalam minimarket 212 |
| 4. | user | Merupakan entitas yang berisi informasi user yang melayani penjualan di minimarket 212 | Merupakan user yang melayani penjualan di minimarket 212 |

| | | | |
|----|----------|---|-------------------------------------|
| 5. | kategori | Merupakan entitas yang berisi informasi kategori dari data barang . | Merupakan kategori sampling barang. |
|----|----------|---|-------------------------------------|

4. membuat tabel asosiasi atribut dengan entitas

| n o | Nama entitas | atribut | Deskripsi | Tipe data dan ukuran | multi value | nul l |
|-----|--------------|------------------------|---------------------------------|----------------------|-------------|-------|
| 1 | jual | jual_n ofak | Nomer faktur jual | VARC HAR(11) | No | No |
| | | jual_tanggal | Tanggal penjualan | TIMESTAMP | Yes | Yes |
| | | jual_total | Total dari hasil penjualan | DOUBLE | Yes | Yes |
| | | Jual_kembali an | Kembali an dari total penjualan | DOUBLE | Yes | Yes |
| | | Jual_u ser_id | Id user penjualan | INT(11) | No | Yes |
| | | Jual_k eteran gan | Keterangan penjualan | VARC HAR(20) | Yes | Yes |
| 2 | Detail jual | d_jual_id | Detail nomer id jual | INT(11) | No | No |
| | | d_jual_nofak | Detail nomer faktur jual | VARC HAR(11) | Yes | Yes |
| | | d_jual_baran g_id | Detail id jual barang | VARC HAR(15) | Yes | Yes |
| | | d_jual_baran g_nam a | Detail nama barang jual | VARC HAR(150) | No | Yes |
| | | d_jual_baran g_satu an | Detail satuan barang jual | VARC HAR(150) | No | Yes |
| | | d_jual_baran g_harp ok | Detail barang jual harga pokok | DOUBLE | No | Yes |
| | | d_jual_baran g_harj ul | Detail barang harga jual | DOUBLE | No | Yes |
| | | d_jual_qty | detail quantity/ | INT(11) | No | Yes |

| | | | | | | |
|----|----------|----------------------------|---|------------------|-----|-----|
| | | | kuantitas jual | | | |
| | | d_jual _diskon | Detail diskon jual | DOUBL E | No | Yes |
| | | d_jual _total | Detail total jual | DOUBL E | No | Yes |
| 3. | Barang | barang _id | Nomer id barang | VARCH AR(15) | No | No |
| | | barang _nama | Nama barang | VARCH AR(150) | No | Yes |
| | | barang _satuan | Satuan barang | VARCH AR(30) | No | Yes |
| | | barang _harpok | Harga pokok barang | DOUBL E | No | Yes |
| | | barang _harjul | Harga jual barang | DOUBL E | No | Yes |
| | | barang _harjul_grosir | Harga jual grosir | DOUBL E | No | Yes |
| | | barang _stok | stok barang | INT(11) | No | Yes |
| | | barang _min_stok | Stok barang minimu m | INT(11) | No | Yes |
| | | barang _tgl_input | Tanggal masuk/in put barang | TIMEST AMPS | Yes | Yes |
| | | barang _tgl_last_update | Tanggal update terakhir barang | DATETI ME | Yes | Yes |
| | | barang _kategori_id | Nomer Kategori id barang | INT(11) | Yes | Yes |
| | | barang _user_id | Id user | INT(11) | No | Yes |
| 4. | User | user_id | Id user | INT(11) | No | No |
| | | user_nama | Nama user | VARCH AR(11) | No | Yes |
| | | user_username | Username user | VARCH AR(11) | No | Yes |
| | | user_password | Password user | VARCH AR(35) | No | Yes |
| | | user_level | Level/tin gkatan user | VARCH AR(2) | No | Yes |
| | | user_status | Status user | VARCH AR(11) | No | Yes |
| 5. | kategori | kategori_id | Nomer id kategori | INT(11) | No | No |
| | | kategori_nama | Nama kategori | VARCH AR(35) | No | Yes |

5. syntax sql pembuatan database.

/*

Navicat Premium Data Transfer

Source Server : MYSQL_XAMPP

Source Server Type : MySQL

Source Server Version : 100411

Source Host : localhost:3306

Source Schema : db_penjualan

Target Server Type : MySQL

Target Server Version : 100411

File Encoding : 65001

Date: 05/07/2020 00:24:39

*/

SET NAMES utf8mb4;

SET FOREIGN_KEY_CHECKS = 0;

-- Table structure for tbl_barang

DROP TABLE IF EXISTS `tbl_barang`; // **Membuat tabel barang**

CREATE TABLE `tbl_barang` (

`barang_id` varchar(15) CHARACTER SET latin1

COLLATE latin1_swedish_ci NOT NULL,

`barang_nama` varchar(150) CHARACTER SET latin1

COLLATE latin1_swedish_ci NULL DEFAULT NULL,

`barang_satuan` varchar(30) CHARACTER SET latin1

COLLATE latin1_swedish_ci NULL DEFAULT NULL,

`barang_harpok` double NULL DEFAULT NULL,

`barang_harjul` double NULL DEFAULT NULL,

`barang_harjul_grosir` double NULL DEFAULT NULL,

`barang_stok` int NULL DEFAULT 0,

`barang_min_stok` int NULL DEFAULT 0,

`barang_tgl_input` timestamp(0) NULL DEFAULT

current_timestamp(0),

`barang_tgl_last_update` datetime(0) NULL DEFAULT

NULL,

`barang_kategori_id` int NULL DEFAULT NULL,

`barang_user_id` int NULL DEFAULT NULL, //

entitas-entitas pada tabel barang

```

PRIMARY KEY (`barang_id`) USING BTREE, //
menggunakan PK pada entitas barang_id
INDEX `barang_user_id`(`barang_user_id`) USING
BTREE, // menggunakan struktur data B-Tree sebagai
peng index an
INDEX `barang_kategori_id`(`barang_kategori_id`)
USING BTREE,
CONSTRAINT `tbl_barang_ibfk_1` FOREIGN KEY
(`barang_user_id`) REFERENCES `tbl_user` (`user_id`)
ON DELETE RESTRICT ON UPDATE CASCADE, //
aturan sebuah tabel yaitu menggunakan foreign key
yang artinya sebagai rujukan contoh dalam script
tabel barang_user_id dikaitkan dengan tabel user
sebagai keterkaitan nilai dan ada keterangan on
delete...dan on update... yang artinya apabila suatu
nilai tabel barang_user_id dirubah, tabel_user pun
juga otomatis terubah
CONSTRAINT `tbl_barang_ibfk_2` FOREIGN KEY
(`barang_kategori_id`) REFERENCES `tbl_kategori`
(`kategori_id`) ON DELETE RESTRICT ON UPDATE
CASCADE
) ENGINE = InnoDB CHARACTER SET = latin1
COLLATE = latin1_swedish_ci ROW_FORMAT =
Dynamic;

-- -----
-- Table structure for tbl_beli
-- -----

DROP TABLE IF EXISTS `tbl_beli`;
CREATE TABLE `tbl_beli` ( // Membuat tabel beli
  `beli_nofak` varchar(15) CHARACTER SET latin1
  COLLATE latin1_swedish_ci NULL DEFAULT NULL,
  `beli_tanggal` date NULL DEFAULT NULL,
  `beli_supplier_id` int NULL DEFAULT NULL,
  `beli_user_id` int NULL DEFAULT NULL,
  `beli_kode` varchar(15) CHARACTER SET latin1
  COLLATE latin1_swedish_ci NOT NULL, // entitas-
entitas pada tabel beli
  PRIMARY KEY (`beli_kode`) USING BTREE, , //
menggunakan PK pada entitas beli_kode

```

```

INDEX `beli_user_id`(`beli_user_id`) USING BTREE,
INDEX `beli_supplier_id`(`beli_supplier_id`) USING
BTREE,
INDEX `beli_id`(`beli_kode`) USING BTREE, //
menggunakan struktur data B-Tree sebagai peng
index an
CONSTRAINT `tbl_beli_ibfk_1` FOREIGN KEY
(`beli_user_id`) REFERENCES `tbl_user` (`user_id`) ON
DELETE RESTRICT ON UPDATE CASCADE, //
aturan sebuah tabel yaitu menggunakan foreign key
yang artinya sebagai rujukan contoh dalam script
tabel beli_user_id dikaitkan dengan tabel user sebagai
keterkaitan nilai dan ada keterangan on delete...dan
on update... yang artinya apabila suatu nilai tabel
beli_user_id dirubah, tabel_user pun juga otomatis
terubah
CONSTRAINT `tbl_beli_ibfk_2` FOREIGN KEY
(`beli_supplier_id`) REFERENCES `tbl_supplier`
(`supplier_id`) ON DELETE RESTRICT ON UPDATE
CASCADE
) ENGINE = InnoDB CHARACTER SET = latin1
COLLATE = latin1_swedish_ci ROW_FORMAT =
Dynamic;

-- -----
-- Records of tbl_beli
-- -----

-- -----
-- Table structure for tbl_detail_beli
-- -----

DROP TABLE IF EXISTS `tbl_detail_beli`;
CREATE TABLE `tbl_detail_beli` ( ( // Membuat tabel
detail beli
  `d_beli_id` int NOT NULL AUTO_INCREMENT,
  `d_beli_nofak` varchar(15) CHARACTER SET latin1
  COLLATE latin1_swedish_ci NULL DEFAULT NULL,

```

```
`d_beli_barang_id` varchar(15) CHARACTER SET
latin1 COLLATE latin1_swedish_ci NULL DEFAULT
NULL,
`d_beli_harga` double NULL DEFAULT NULL,
`d_beli_jumlah` int NULL DEFAULT NULL,
`d_beli_total` double NULL DEFAULT NULL,
`d_beli_kode` varchar(15) CHARACTER SET latin1
COLLATE latin1_swedish_ci NULL DEFAULT NULL,
// entitas-entitas pada tabel detail beli
```

PRIMARY KEY (`d_beli_id`) USING BTREE, //
**menggunakan PK pada entitas beli_kode dan
menggunakan struktur data B-Tree sebagai Langkah
pengindex an**

```
INDEX `d_beli_barang_id`(`d_beli_barang_id`) USING
BTREE,
INDEX `d_beli_nofak`(`d_beli_nofak`) USING BTREE,
INDEX `d_beli_kode`(`d_beli_kode`) USING BTREE,
CONSTRAINT `tbl_detail_beli_ibfk_1` FOREIGN
KEY (`d_beli_barang_id`) REFERENCES `tbl_barang`
(`barang_id`) ON DELETE RESTRICT ON UPDATE
CASCADE, // aturan sebuah tabel yaitu menggunakan
foreign key yang artinya sebagai rujukan contoh
dalam script tabel d_beli_barang_id dikaitkan dengan
tbl_barang sebagai keterkaitan nilai dan ada
keterangan on delete...dan on update... yang artinya
apabila suatu nilai tabel d_beli_barang_id dirubah,
tbl_barang pun juga otomatis terubah
```

```
CONSTRAINT `tbl_detail_beli_ibfk_2` FOREIGN
KEY (`d_beli_kode`) REFERENCES `tbl_beli`
(`beli_kode`) ON DELETE RESTRICT ON UPDATE
CASCADE
) ENGINE = InnoDB AUTO_INCREMENT = 1
CHARACTER SET = latin1 COLLATE =
latin1_swedish_ci ROW_FORMAT = Dynamic;
```

```
-- -----
-- Records of tbl_detail_beli
-- -----
```

```
-- -----
```

```
-- Table structure for tbl_detail_jual
```

```
-- -----
```

```
DROP TABLE IF EXISTS `tbl_detail_jual`;
CREATE TABLE `tbl_detail_jual` ( // Membuat tabel  

detail beli
`d_jual_id` int NOT NULL AUTO_INCREMENT,
`d_jual_nofak` varchar(15) CHARACTER SET latin1
COLLATE latin1_swedish_ci NULL DEFAULT NULL,
`d_jual_barang_id` varchar(15) CHARACTER SET
latin1 COLLATE latin1_swedish_ci NULL DEFAULT
NULL,
`d_jual_barang_nama` varchar(150) CHARACTER SET
latin1 COLLATE latin1_swedish_ci NULL DEFAULT
NULL,
`d_jual_barang_satuan` varchar(30) CHARACTER SET
latin1 COLLATE latin1_swedish_ci NULL DEFAULT
NULL,
`d_jual_barang_harpok` double NULL DEFAULT
NULL,
`d_jual_barang_harjul` double NULL DEFAULT
NULL,
`d_jual_qty` int NULL DEFAULT NULL,
`d_jual_diskon` double NULL DEFAULT NULL,
`d_jual_total` double NULL DEFAULT NULL, //
entitas-entitas pada tabel detail beli
PRIMARY KEY (`d_jual_id`) USING BTREE, //
menggunakan PK pada entitas d_jual_id dan  

menggunakan struktur data B-Tree sebagai Langkah  

pengindex an
INDEX `d_jual_barang_id`(`d_jual_barang_id`) USING
BTREE,
INDEX `d_jual_nofak`(`d_jual_nofak`) USING BTREE,
CONSTRAINT `tbl_detail_jual_ibfk_1` FOREIGN
KEY (`d_jual_barang_id`) REFERENCES `tbl_barang`
(`barang_id`) ON DELETE RESTRICT ON UPDATE
CASCADE, // aturan sebuah tabel yaitu menggunakan  

foreign key yang artinya sebagai rujukan contoh  

dalam script tabel d_jual_barang_id dikaitkan dengan
```

tbl_barang sebagai keterkaitan nilai dan ada keterangan on delete...dan on update... yang artinya apabila suatu nilai tabel **d_jual_barang_id** dirubah, nilai **barang_id** pun juga otomatis berubah

```
CONSTRAINT `tbl_detail_jual_ibfk_2` FOREIGN
KEY (`d_jual_nofak`) REFERENCES `tbl_jual`
(`jual_nofak`) ON DELETE RESTRICT ON UPDATE
CASCADE
) ENGINE = InnoDB AUTO_INCREMENT = 33
CHARACTER SET = latin1 COLLATE =
latin1_swedish_ci ROW_FORMAT = Dynamic;
```

```
--
-- Records of tbl_detail_jual
--
```

```
INSERT INTO `tbl_detail_jual` VALUES (27,
'010720000001', 'BR000024', 'Good Time', 'pcs', 55000,
59250, 1, 0, 59250);
INSERT INTO `tbl_detail_jual` VALUES (28,
'020720000002', 'BR000024', 'Good Time', 'pcs', 55000,
59250, 1, 0, 59250);
INSERT INTO `tbl_detail_jual` VALUES (29,
'020720000003', 'BR000007', 'patatoz barbeque', 'pcs',
43000, 44800, 1, 0, 44800);
INSERT INTO `tbl_detail_jual` VALUES (30,
'020720000003', 'BR000002', 'Leo Kripik Kentang Sapi
Panggang 150gr', 'pcs', 475000, 550000, 1, 0, 550000);
INSERT INTO `tbl_detail_jual` VALUES (31,
'040720000001', 'BR000024', 'Good Time', 'pcs', 4750,
9500, 9, 0, 85500);
INSERT INTO `tbl_detail_jual` VALUES (32,
'040720000001', 'BR000001', 'Leo Kripik Kentang
Rumput Laut. 150gr', 'pcs', 1500, 3000, 10, 0, 30000); //
```

Memasukkan isi tabel pada tabel detail_jual

```
--
--
-- Table structure for tbl_detail_beli
--
```

```
DROP TABLE IF EXISTS `tbl_detail_beli`;
```

CREATE TABLE `tbl_detail_beli` ((// **Membuat tabel detail beli**

```
`d_beli_id` int NOT NULL AUTO_INCREMENT,
`d_beli_nofak` varchar(15) CHARACTER SET latin1
COLLATE latin1_swedish_ci NULL DEFAULT NULL,
`d_beli_barang_id` varchar(15) CHARACTER SET
latin1 COLLATE latin1_swedish_ci NULL DEFAULT
NULL,
`d_beli_harga` double NULL DEFAULT NULL,
`d_beli_jumlah` int NULL DEFAULT NULL,
`d_beli_total` double NULL DEFAULT NULL,
`d_beli_kode` varchar(15) CHARACTER SET latin1
COLLATE latin1_swedish_ci NULL DEFAULT NULL,
// entitas-entitas pada tabel detail beli
```

PRIMARY KEY (`d_beli_id`) USING BTREE, //
menggunakan PK pada entitas beli_kode dan menggunakan struktur data B-Tree sebagai Langkah pengindex an

```
INDEX `d_beli_barang_id` (`d_beli_barang_id`) USING
BTREE,
INDEX `d_beli_nofak` (`d_beli_nofak`) USING BTREE,
INDEX `d_beli_kode` (`d_beli_kode`) USING BTREE,
CONSTRAINT `tbl_detail_beli_ibfk_1` FOREIGN
KEY (`d_beli_barang_id`) REFERENCES `tbl_barang`
(`barang_id`) ON DELETE RESTRICT ON UPDATE
CASCADE, // aturan sebuah tabel yaitu menggunakan
foreign key yang artinya sebagai rujukan contoh
dalam script tabel d_beli_barang_id dikaitkan dengan
tbl_barang sebagai keterkaitan nilai dan ada
keterangan on delete...dan on update... yang artinya
apabila suatu nilai tabel d_beli_barang_id dirubah,
tbl_barang pun juga otomatis berubah
```

```
CONSTRAINT `tbl_detail_beli_ibfk_2` FOREIGN
KEY (`d_beli_kode`) REFERENCES `tbl_beli`
(`beli_kode`) ON DELETE RESTRICT ON UPDATE
CASCADE
```



```

) ENGINE = InnoDB AUTO_INCREMENT = 1
CHARACTER SET = latin1 COLLATE =
latin1_swedish_ci ROW_FORMAT = Dynamic;

-- -----
-- Records of tbl_detail_beli
-- -----

-- -----
-- Table structure for tbl_detail_jual
-- -----

DROP TABLE IF EXISTS `tbl_detail_jual`;
CREATE TABLE `tbl_detail_jual` ( // Membuat tabel detail beli
  `d_jual_id` int NOT NULL AUTO_INCREMENT,
  `d_jual_nofak` varchar(15) CHARACTER SET latin1
  COLLATE latin1_swedish_ci NULL DEFAULT NULL,
  `d_jual_barang_id` varchar(15) CHARACTER SET
  latin1 COLLATE latin1_swedish_ci NULL DEFAULT
  NULL,
  `d_jual_barang_nama` varchar(150) CHARACTER SET
  latin1 COLLATE latin1_swedish_ci NULL DEFAULT
  NULL,
  `d_jual_barang_satuan` varchar(30) CHARACTER SET
  latin1 COLLATE latin1_swedish_ci NULL DEFAULT
  NULL,
  `d_jual_barang_harpok` double NULL DEFAULT
  NULL,
  `d_jual_barang_harjul` double NULL DEFAULT
  NULL,
  `d_jual_qty` int NULL DEFAULT NULL,
  `d_jual_diskon` double NULL DEFAULT NULL,
  `d_jual_total` double NULL DEFAULT NULL, //
entitas-entitas pada tabel detail beli
  PRIMARY KEY (`d_jual_id`) USING BTREE, //
menggunakan PK pada entitas d_jual_id dan
menggunakan struktur data B-Tree sebagai Langkah
pengindex an
  INDEX `d_jual_barang_id` (`d_jual_barang_id`) USING
  BTREE,

```

```

  INDEX `d_jual_nofak` (`d_jual_nofak`) USING BTREE,
  CONSTRAINT `tbl_detail_jual_ibfk_1` FOREIGN
  KEY (`d_jual_barang_id`) REFERENCES `tbl_barang`
  (`barang_id`) ON DELETE RESTRICT ON UPDATE
  CASCADE, // aturan sebuah tabel yaitu menggunakan
foreign key yang artinya sebagai rujukan contoh
dalam script tabel d_jual_barang_id dikaitkan dengan
tbl_barang sebagai keterkaitan nilai dan ada
keterangan on delete...dan on update... yang artinya
apabila suatu nilai tabel d_jual_barang_id dirubah,
nilai barang_id pun juga otomatis terubah
  CONSTRAINT `tbl_detail_jual_ibfk_2` FOREIGN
  KEY (`d_jual_nofak`) REFERENCES `tbl_jual`
  (`jual_nofak`) ON DELETE RESTRICT ON UPDATE
  CASCADE
) ENGINE = InnoDB AUTO_INCREMENT = 33
CHARACTER SET = latin1 COLLATE =
latin1_swedish_ci ROW_FORMAT = Dynamic;

-- -----
-- Records of tbl_detail_jual
-- -----

INSERT INTO `tbl_detail_jual` VALUES (27,
'010720000001', 'BR000024', 'Good Time', 'pcs', 55000,
59250, 1, 0, 59250);
INSERT INTO `tbl_detail_jual` VALUES (28,
'020720000002', 'BR000024', 'Good Time', 'pcs', 55000,
59250, 1, 0, 59250);
INSERT INTO `tbl_detail_jual` VALUES (29,
'020720000003', 'BR000007', 'patatoz barbeque', 'pcs',
43000, 44800, 1, 0, 44800);
INSERT INTO `tbl_detail_jual` VALUES (30,
'020720000003', 'BR000002', 'Leo Kripik Kentang Sapi
Panggang 150gr', 'pcs', 475000, 550000, 1, 0, 550000);
INSERT INTO `tbl_detail_jual` VALUES (31,
'040720000001', 'BR000024', 'Good Time', 'pcs', 4750,
9500, 9, 0, 85500);
INSERT INTO `tbl_detail_jual` VALUES (32,
'040720000001', 'BR000001', 'Leo Kripik Kentang

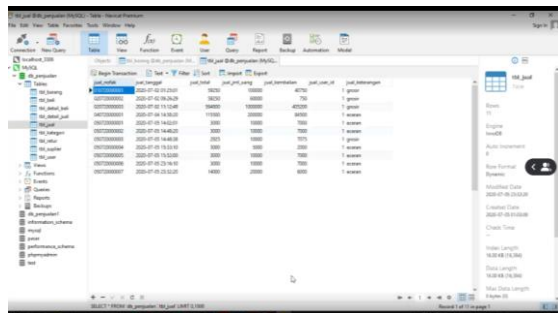
```

Rumput Laut. 150gr', 'pcs', 1500, 3000, 10, 0, 30000); //

Memasukkan isi tabel pada tabel detail_jual

6. HASIL SQL

1. TABEL JUAL



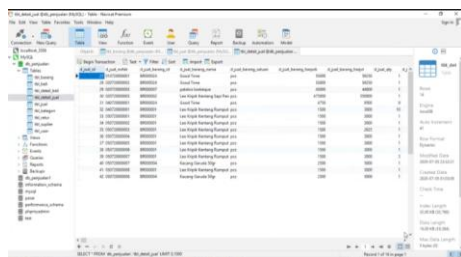
The screenshot shows the MySQL Workbench interface with the 'jual' table selected. The table structure is as follows:

| Field | Type | Length | Null | Key | Extra |
|--------------|------|--------|------|---------|-------------------------------------|
| id_jual | INT | 11 | NO | PRIMARY | |
| id_barang | INT | 11 | NO | FOREIGN | REFERENCES barang (id_barang) |
| id_pelanggan | INT | 11 | NO | FOREIGN | REFERENCES pelanggan (id_pelanggan) |
| tanggal_jual | DATE | 10 | NO | | |
| jumlah_jual | INT | 11 | NO | | |
| harga_jual | INT | 11 | NO | | |
| diskon | INT | 11 | NO | | |
| total_harga | INT | 11 | NO | | |

The data view shows the following records:

| id_jual | id_barang | id_pelanggan | tanggal_jual | jumlah_jual | harga_jual | diskon | total_harga |
|---------|-----------|--------------|--------------|-------------|------------|--------|-------------|
| 1 | 1 | 1 | 2020-07-10 | 1500 | 3000 | 10 | 27000 |
| 2 | 2 | 2 | 2020-07-11 | 1500 | 3000 | 10 | 27000 |
| 3 | 3 | 3 | 2020-07-12 | 1500 | 3000 | 10 | 27000 |
| 4 | 4 | 4 | 2020-07-13 | 1500 | 3000 | 10 | 27000 |
| 5 | 5 | 5 | 2020-07-14 | 1500 | 3000 | 10 | 27000 |
| 6 | 6 | 6 | 2020-07-15 | 1500 | 3000 | 10 | 27000 |
| 7 | 7 | 7 | 2020-07-16 | 1500 | 3000 | 10 | 27000 |
| 8 | 8 | 8 | 2020-07-17 | 1500 | 3000 | 10 | 27000 |
| 9 | 9 | 9 | 2020-07-18 | 1500 | 3000 | 10 | 27000 |
| 10 | 10 | 10 | 2020-07-19 | 1500 | 3000 | 10 | 27000 |

2. Tabel detail_jual

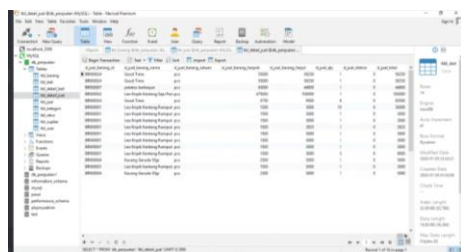


The screenshot shows the MySQL Workbench interface with the 'detail_jual' table selected. The table structure is as follows:

| Field | Type | Length | Null | Key | Extra |
|--------------------|------|--------|------|---------|-------------------------------|
| id_detail_jual | INT | 11 | NO | PRIMARY | |
| id_jual | INT | 11 | NO | FOREIGN | REFERENCES jual (id_jual) |
| id_barang | INT | 11 | NO | FOREIGN | REFERENCES barang (id_barang) |
| jumlah_barang | INT | 11 | NO | | |
| harga_barang | INT | 11 | NO | | |
| diskon_barang | INT | 11 | NO | | |
| total_harga_barang | INT | 11 | NO | | |

The data view shows the following records:

| id_detail_jual | id_jual | id_barang | jumlah_barang | harga_barang | diskon_barang | total_harga_barang |
|----------------|---------|-----------|---------------|--------------|---------------|--------------------|
| 1 | 1 | 1 | 1500 | 3000 | 10 | 27000 |
| 2 | 2 | 2 | 1500 | 3000 | 10 | 27000 |
| 3 | 3 | 3 | 1500 | 3000 | 10 | 27000 |
| 4 | 4 | 4 | 1500 | 3000 | 10 | 27000 |
| 5 | 5 | 5 | 1500 | 3000 | 10 | 27000 |
| 6 | 6 | 6 | 1500 | 3000 | 10 | 27000 |
| 7 | 7 | 7 | 1500 | 3000 | 10 | 27000 |
| 8 | 8 | 8 | 1500 | 3000 | 10 | 27000 |
| 9 | 9 | 9 | 1500 | 3000 | 10 | 27000 |
| 10 | 10 | 10 | 1500 | 3000 | 10 | 27000 |



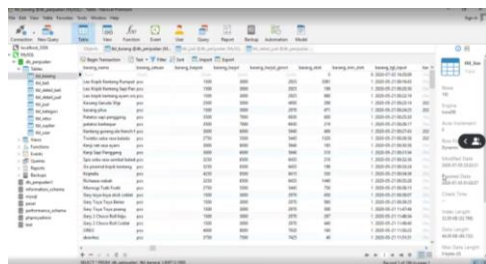
The screenshot shows the MySQL Workbench interface with the 'barang' table selected. The table structure is as follows:

| Field | Type | Length | Null | Key | Extra |
|--------------------|---------|--------|------|---------|-------|
| id_barang | INT | 11 | NO | PRIMARY | |
| nama_barang | VARCHAR | 255 | YES | | |
| jenis_barang | VARCHAR | 255 | YES | | |
| berat_barang | INT | 11 | NO | | |
| harga_barang | INT | 11 | NO | | |
| diskon_barang | INT | 11 | NO | | |
| total_harga_barang | INT | 11 | NO | | |

The data view shows the following records:

| id_barang | nama_barang | jenis_barang | berat_barang | harga_barang | diskon_barang | total_harga_barang |
|-----------|-------------|--------------|--------------|--------------|---------------|--------------------|
| 1 | Rumput Laut | 150gr' | 1500 | 3000 | 10 | 27000 |
| 2 | Rumput Laut | 150gr' | 1500 | 3000 | 10 | 27000 |
| 3 | Rumput Laut | 150gr' | 1500 | 3000 | 10 | 27000 |
| 4 | Rumput Laut | 150gr' | 1500 | 3000 | 10 | 27000 |
| 5 | Rumput Laut | 150gr' | 1500 | 3000 | 10 | 27000 |
| 6 | Rumput Laut | 150gr' | 1500 | 3000 | 10 | 27000 |
| 7 | Rumput Laut | 150gr' | 1500 | 3000 | 10 | 27000 |
| 8 | Rumput Laut | 150gr' | 1500 | 3000 | 10 | 27000 |
| 9 | Rumput Laut | 150gr' | 1500 | 3000 | 10 | 27000 |
| 10 | Rumput Laut | 150gr' | 1500 | 3000 | 10 | 27000 |

3. tabel barang

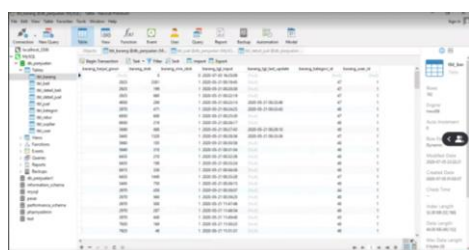


The screenshot shows the MySQL Workbench interface with the 'barang' table selected. The table structure is as follows:

| Field | Type | Length | Null | Key | Extra |
|--------------------|---------|--------|------|---------|-------|
| id_barang | INT | 11 | NO | PRIMARY | |
| nama_barang | VARCHAR | 255 | YES | | |
| jenis_barang | VARCHAR | 255 | YES | | |
| berat_barang | INT | 11 | NO | | |
| harga_barang | INT | 11 | NO | | |
| diskon_barang | INT | 11 | NO | | |
| total_harga_barang | INT | 11 | NO | | |

The data view shows the following records:

| id_barang | nama_barang | jenis_barang | berat_barang | harga_barang | diskon_barang | total_harga_barang |
|-----------|-------------|--------------|--------------|--------------|---------------|--------------------|
| 1 | Rumput Laut | 150gr' | 1500 | 3000 | 10 | 27000 |
| 2 | Rumput Laut | 150gr' | 1500 | 3000 | 10 | 27000 |
| 3 | Rumput Laut | 150gr' | 1500 | 3000 | 10 | 27000 |
| 4 | Rumput Laut | 150gr' | 1500 | 3000 | 10 | 27000 |
| 5 | Rumput Laut | 150gr' | 1500 | 3000 | 10 | 27000 |
| 6 | Rumput Laut | 150gr' | 1500 | 3000 | 10 | 27000 |
| 7 | Rumput Laut | 150gr' | 1500 | 3000 | 10 | 27000 |
| 8 | Rumput Laut | 150gr' | 1500 | 3000 | 10 | 27000 |
| 9 | Rumput Laut | 150gr' | 1500 | 3000 | 10 | 27000 |
| 10 | Rumput Laut | 150gr' | 1500 | 3000 | 10 | 27000 |



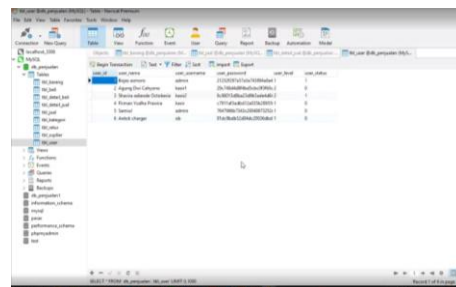
The screenshot shows the MySQL Workbench interface with the 'barang' table selected. The table structure is as follows:

| Field | Type | Length | Null | Key | Extra |
|--------------------|---------|--------|------|---------|-------|
| id_barang | INT | 11 | NO | PRIMARY | |
| nama_barang | VARCHAR | 255 | YES | | |
| jenis_barang | VARCHAR | 255 | YES | | |
| berat_barang | INT | 11 | NO | | |
| harga_barang | INT | 11 | NO | | |
| diskon_barang | INT | 11 | NO | | |
| total_harga_barang | INT | 11 | NO | | |

The data view shows the following records:

| id_barang | nama_barang | jenis_barang | berat_barang | harga_barang | diskon_barang | total_harga_barang |
|-----------|-------------|--------------|--------------|--------------|---------------|--------------------|
| 1 | Rumput Laut | 150gr' | 1500 | 3000 | 10 | 27000 |
| 2 | Rumput Laut | 150gr' | 1500 | 3000 | 10 | 27000 |
| 3 | Rumput Laut | 150gr' | 1500 | 3000 | 10 | 27000 |
| 4 | Rumput Laut | 150gr' | 1500 | 3000 | 10 | 27000 |
| 5 | Rumput Laut | 150gr' | 1500 | 3000 | 10 | 27000 |
| 6 | Rumput Laut | 150gr' | 1500 | 3000 | 10 | 27000 |
| 7 | Rumput Laut | 150gr' | 1500 | 3000 | 10 | 27000 |
| 8 | Rumput Laut | 150gr' | 1500 | 3000 | 10 | 27000 |
| 9 | Rumput Laut | 150gr' | 1500 | 3000 | 10 | 27000 |
| 10 | Rumput Laut | 150gr' | 1500 | 3000 | 10 | 27000 |

4. tabel user



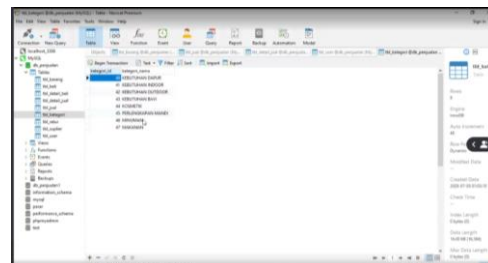
The screenshot shows the MySQL Workbench interface with the 'user' table selected. The table structure is as follows:

| Field | Type | Length | Null | Key | Extra |
|-----------|---------|--------|------|---------|-------|
| id_user | INT | 11 | NO | PRIMARY | |
| nama_user | VARCHAR | 255 | YES | | |
| password | VARCHAR | 255 | YES | | |
| role | VARCHAR | 255 | YES | | |

The data view shows the following records:

| id_user | nama_user | password | role |
|---------|-----------|----------|-------|
| 1 | Admin | admin | Admin |
| 2 | Admin | admin | Admin |
| 3 | Admin | admin | Admin |
| 4 | Admin | admin | Admin |
| 5 | Admin | admin | Admin |
| 6 | Admin | admin | Admin |
| 7 | Admin | admin | Admin |
| 8 | Admin | admin | Admin |
| 9 | Admin | admin | Admin |
| 10 | Admin | admin | Admin |

5. tabel kategori



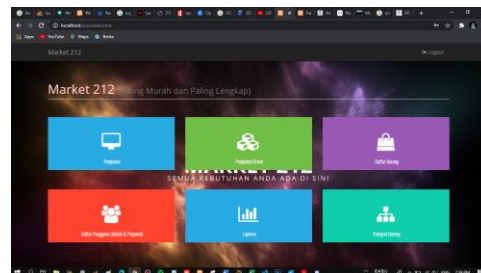
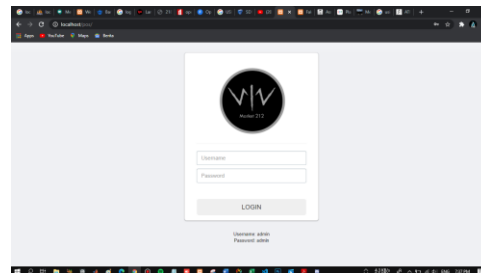
The screenshot shows the MySQL Workbench interface with the 'kategori' table selected. The table structure is as follows:

| Field | Type | Length | Null | Key | Extra |
|---------------|---------|--------|------|---------|-------|
| id_kategori | INT | 11 | NO | PRIMARY | |
| nama_kategori | VARCHAR | 255 | YES | | |

The data view shows the following records:

| id_kategori | nama_kategori |
|-------------|---------------|
| 1 | Rumput Laut |
| 2 | Rumput Laut |
| 3 | Rumput Laut |
| 4 | Rumput Laut |
| 5 | Rumput Laut |
| 6 | Rumput Laut |
| 7 | Rumput Laut |
| 8 | Rumput Laut |
| 9 | Rumput Laut |
| 10 | Rumput Laut |

7.importing data ke website/aplikasi.



4 KESIMPULAN DAN SARAN

4.1 Kesimpulan

Setelah sistem ini diterapkan ada beberapa hal yang dapat diambil sebagai kesimpulan :

menggunakan suatu sistem komputerisasi dalam penjualan barang, maka proses penjualan barang dapat berjalan dengan cepat dan terakurat dibanding dengan manual, hal ini diakibatkan karena mudahnya mengakses informasi data barang melalui sistem komputerisasi. pendataan dengan menggunakan SQL lebih cepat dan akurat hasilnya dalam proses CRUD data ke dalam sistem dan mengurangi adanya human error. kemudian menghasilkan database dengan model

relasional agar menjaga data setiap tabel yang berhubungan dan menjaga setiap entity dan attributes didalamnya.

5 REFERENSI

Kadir, Abdul 2009. *Membuat Aplikasi Web dengan PHP + Database MySQL*. Yogyakarta: Andi Offset

Fresnel, Teddy 2012, *ANALISIS DAN PERANCANGAN SISTEM INFORMASI PENJUALAN MINIMARKET BERINTEGRASI BARCODE READER MENGGUNAKAN PHP, MYSQL DAN JQUERY*.yogyakarta

Saputro, Haris 2012. *MODUL PEMBELAJARAN PRAKTIK BASIS DATA*.

Cek plagiasi

<https://smallseotools.com/view-report/62a0ed2c8a9ccffe603a19e5a920237>

<https://smallseotools.com/view-report/b16aabcb67c880eb752807bf124d899>

<https://smallseotools.com/view-report/a754cf3d06fcd4f4ccab9a77cdd7e9cb>

<https://smallseotools.com/view-report/d18fe29dab24582445599656e09c72e4>

<https://smallseotools.com/view-report/58a36281f9395c1102cd73c93ff8db31>

