MAKALAH

PENGGUNAAN JOIN DALAM DATABASE SQL MENGGUNAKAN MARIADB STUDI KASUS SISTEM INFORMASI PENJUALAN APOTIK

Disusun guna memenuhi salah satu tugas mata kuliah Database Manajemen Sistem Lanjut

Dosen : Gunawan, S.T, M. Kom,. MOS

Disusun Oleh :

Kelompok 1

Isep Lutpi Nur	2113191079
Adistia Ramadhani	2113191084
Dara Atria Ferliandini	2113191098
Alam Nurzaman	2113191108

PROGRAM STUDI TEKNIK INFORMATIKA

UNIVERSITAS SANGGA BUANA YPKP

BANDUNG

2021

KATA PENGANTAR

Puji syukur kehadirat Allah SWT yang telah memberikan rahmat dan hidayah-Nya sehingga kami bisa menyelesaikan tugas makalah yang berjudul ***PENGGUNAAN JOIN DALAM DATABASE SQL MENGGUNAKAN MARIADB STUDIKASUS SISTEM INFORMASI PENJUALAN APOTIK*** ini dengan tepat pada waktunya.

Adapun tujuan dari penulisan makalah ini adalah untuk memenuhi tugas mata kuliah kewarganegaraan. Selain itu, makalah ini juga bertujuan untuk menambah wawasan bagi para pembaca dan juga penulis.

Kami mengucapkan terima kasih kepada pihak yang telah membagi sebagian pengetahuannya sehingga kami dapat menyelesaikan makalah ini.

Kami menyadari, makalah yang kami tulis ini masih jauh dari kata sempurna. Oleh karena itu, kritik dan saran yang membangun akan kami nantikan demi kesempurnaan makalah ini.

Bandung, Oktober 2021

Penulis

DAFTAR ISI

KATA PENGANTAR I
DAFTAR ISI II
BAB I PENDAHULUAN 1
1.1 Latar Belakang 1
1.2 Rumusan Masalah 2
1.3 Batasan Masalah 2
1.4 Tujuan Pembahasan 2
1.5 Pengertian Join Table 2
1.6 Jenis-jenis join table dalam database SQL 2
1.7 Manfaat join table dalam database SQL 6
1.8 Fungsi join table dalam database SQL 6
BAB II STUDY KASUS 7
2.1 Deskripsi kasus yang diambil 7
2.2 Skema Database 7
2.3 SQL Data Definition Language 10
2.4 SQL Data Manipulation Language 15
BAB III PEMBAHASAN STUDY KASUS 19
BAB IV PENUTUP 21
4.1 Kesimpulan 21
DAFTAR PUSTAKA 22

BAB I PENDAHULUAN

1.1 Latar Belakang

Semakin berkembangnya teknologi informasi pada saat ini sangat membantu setiap pekerjaan manusia. Seperti dalam hal pengumpulan data, setiap orang dalam suatu institusi atau perusahaan pasti tidak bisa lepas dari menggunakan DBMS (Database Management System). Dari yang sederhana seperti menggunakan Microsoft Access sampai dengan menggunakan DBMS yang cukup kompleks seperti Oracle. DBMS ini bertujuan untuk mempermudah dalam hal penyimpanan data maupun dalam hal manipulasi data, yang nantinya data tersebut dapat digunakan kembali apabila diperlukan.

Selain teknologi pengumpulan data yang terus berkembang, teknologi penyimpanan data pun terus mengalami peningkatan. Dahulu biasanya suatu media penyimpanan seperti Harddisk mempunyai kapasitas dalam ukuran Giga, tetapi sekarang banyak ditemui kapasitas Harddisk yang sampai pada ukuran Tera. Hal ini sangat membantu suatu sekolah yang akan menyimpan data yang mempunyai ukuran yang cukup besar.

Sekarang ini juga banyak suatu perusahaan, toko, supermarket, apotek dan lainnya menggunakan sebuah sistem informasi. Sistem informasi tersebut dibuat agar memudahkan dalam mengelola informasi di suatu perusahaan maupun lainnya. Sistem informasi–sistem informasi tersebut ada yang berbasis web dan ada yang berbasis desktop. Adanya sistem informasi saat ini pengelolaan dalam suatu perusahaan maupun yang lainnya akan sangat mudah dan sangat cepat. Serta tidak lagi menggunakan sistem secara manual, dan kadang sulit dalam pengelolaannya.

Sebuah apotek pasti membutuhkan suatu sistem informasi yang dapat digunakan untuk mengelola atau memanajemenkan keluar masuknya obat dan pada apotik. Karena apotek juga merupakan salah satu sentra dalam pembuatan 2 sistem informasi. Sistem informasi yang dibutuhkan berupa sistem informasi yang menangani tentang sistem jual beli, keluar masuknya obat, jumlah obat. Apotek tersebut masih menggunakan sistem informasi yang bersifat manual. Dalam informasi yang masih bersifat manual sering terjadi kesalahan maupun kekeliruan dalam pengelolaan keluar masuknya obat tersebut.

1.2 Rumusan Masalah

1. Apa itu join table dalam database SQL?
2. Jenis-Jenis Join table dan perintahnya?
3. Apa manfaat dari Join table dalam database SQL?
4. Cara penyelesaian sebuah kasus menggunakan perintah Join table dalam database SQL?

1.3 Batasan Masalah

Pada makalah ini hanya akan berfokus pada join table dalam database SQL menggunakan DBMS MariaDB Dengan studi kasus penggunaan join dalam database sql menggunakan mariadb studikasus sistem informasi penjualan apotik.

1.4 Tujuan Pembahasan

1. Untuk mengetahui join table dalam database SQL.
2. Untuk mengetahui Jenis-Jenis Join table dan perintahnya.
3. Untuk mengetahui manfaat dari Join table dalam database SQL.
4. Untuk mengetahui cara implementasi perintah join table dalam database SQL untuk memecahkan sebuah kasus.

1.5 Pengertian Join Table

Join adalah penggabungan table yang dilakukan melalui kolom / key tertentu yang memiliki nilai terkait untuk mendapatkan satu set data dengan informasi lengkap.

Lengkap disini artinya kolom data didapatkan dari kolom-kolom hasil join antar table tersebut. Join diperlukan karena perancangan table pada sistem transaksional kebanyakan di-normalisasi, salah satu alasannya untuk menghindari redundansi data.

Pada bahasa SQL, operasi join atau penggabungan antar table adalah operasi dasar database relasional yang sangat penting. Untuk mendukung perancangan database relasional yang baik.

1.6 Jenis-jenis join table dalam database SQL

Di dalam database, ada kalanya kita membutuhkan data dari beberapa tabel yang saling berhubungan. Untuk mendapatkan data dari beberapa tabel tersebut dapat digunakan perintah join pada perintah SQL.

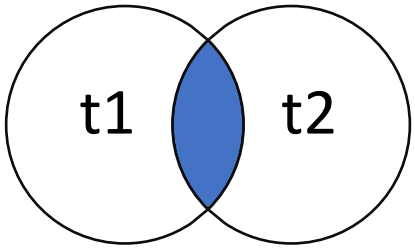
Pada MySQL, perintah join ada beberapa macam yaitu:

2.1.Inner Join

inner join mungkin tipe join yang paling banyak dipakai. ***inner join*** mengembalikan baris-baris dari dua tabel atau lebih yang memenuhi syarat.

Misalkan kita menggunakan klausa ***inner join*** untuk mengambil data dari dua table t1 dan t2, bisa di ilustarasikan menggunakan sintaks ***inner join*** dibawah ini:

```
select list_kolom
from t1
inner join t2 on kondisi_join;
```



Dalam sintaks diatas, klausa ***inner join*** membandingkan setiap baris tabel ***t1*** dengan baris tabel ***t2***. Jika kondisi nya sama dan mengembalikan nilai true (dalam klausa ***join_condition***) maka baris tabel bisa diambil atau ditampilkan. Seperti diagram ilustrasi disamping.

Biasanya join table menggunakan operator sama dengan (=) sebagai pembanding antara kolom di tabel pertama (***t1***) primary key (Kunci utama) dan kolom tabel kedua (***b2***) foreign key (kunci tamu).

```
select list_kolom
from t1
inner join t2 on t1.kolom1 = t2.kolom1;
```

Jika kolom untuk pembanding mempunyai nama yang sama, maka kita bisa menggunakan klausa ***using*** contoh:

```
select list_kolom
from t1
inner join t2 using (kolom1);
```

Untuk join tabel lebih dari dua tabel, kita bisa menambahkan klausa ***inner join*** yang ditunjukan dalam sintaks dibawah berikut:

```
select list_kolom
from t1
inner join t2 on kondisi_join1
inner join t3 on kondisi_join2
```

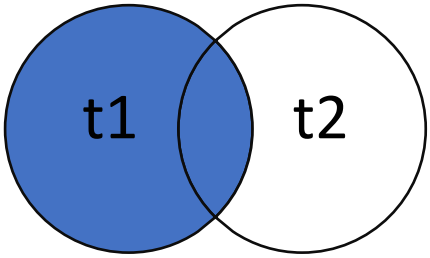
2.2.Left Join

Left join Adalah Relasi Antar Table, biasanya Digunakan untuk menghasilkan baris data dari tabel kiri (nama tabel pertama/ Tabel Utama) yang tidak ada pasangan/Tidak Berelasi datanya pada tabel kanan (nama tabel kedua).

Atau Left Join merupakan penggabungan tabel dimana data akan ditampilkan secara keseluruhan pada tabel pertama (kiri) namun record pada tabel kedua (kanan) yang kosong akan ditampilkan dengan isi NULL.

Misalkan kita menggunakan klausa ***left join*** untuk mengambil data dari dua table t1 dan t2, bisa di ilustarasikan menggunakan sintaks ***left join*** dibawah ini:

```
select list_kolom
from t1
left join t2 on kondisi_join;
```



Dalam ***left join*** dimulai dengan mengambil data dari kiri (left) tabel t1. kemudian bandingkan setiap baris t1 dengan setiap baris t2.

Jika kedua baris sama kondisi join akan mengembalikan nilai true, tetapi jika di tabel t2 kanan tidak baris nya tidak sama maka akan tetap diambil dan diganti dengan nilai null.

Biasanya join table menggunakan operator sama dengan (=) sebagai pembanding antara kolom di tabel pertama (***t1***) primary key (Kunci utama) dan kolom tabel kedua (***b2***) foreign key (kunci tamu).

```
select list_kolom
from t1
left join t2 on t1.kolom1 = t2.kolom1;
```

Jika kolom untuk pembanding mempunyai nama yang sama, maka kita bisa menggunakan klausa ***using*** contoh:

```
select list_kolom
from t1
left join t2 using (kolom1);
```

Untuk join tabel lebih dari dua tabel, kita bisa menambahkan klausa ***left join*** yang ditunjukan dalam sintaks dibawah berikut:

```
select list_kolom
from t1
left join t2 on kondisi_join1
left join t3 on kondisi_join2
```

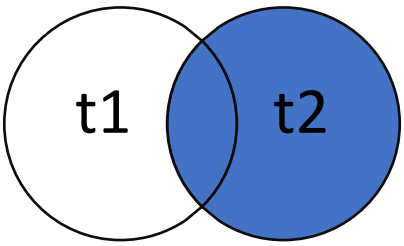
2.3.Right Join

RIGHT JOIN digunakan untuk menghasilkan baris data dari tabel kanan (nama tabel kedua/ Tabel Utama) yang tidak ada pasangan datanya/ Tidak Berelasi pada tabel kiri (nama tabel pertama).

Atau Right Join memiliki fungsi yang bertolak belakang dengan left join, dimana right join akan menampilkan data secara keseluruhan pada tabel kedua (kanan), namun NULL pada tabel pertama (kiri).

Misalkan kita menggunakan klausa ***left join*** untuk mengambil data dari dua table t1 dan t2, bisa di ilustarasikan menggunakan sintaks ***left join*** dibawah ini:

```
select list_kolom
from t1
right join t2 on kondisi_join;
```



Dalam ***right join*** dimulai dengan mengambil data dari kanan (right) tabel t2. kemudian banding kansetiap baris t1 dengan setiap baris t2.

Jika kedua baris sama kondisi join akan mengembalikan nilai true, tetapi jika di tabel t2 kanan tidak baris nya tidak sama maka akan tetap diambil dan diganti dengan nilai null.

Biasanya join table menggunakan operator sama dengan (=) sebagai pembanding antara kolom di tabel pertama (***t1***) primary key (Kunci utama) dan kolom tabel kedua (***b2***) foreign key (kunci tamu).

```
select list_kolom
from t1
right join t2 on t1.kolom1 = t2.kolom1;
```

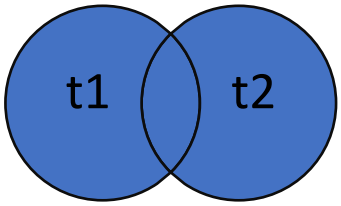
Jika kolom untuk pembanding mempunyai nama yang sama, maka kita bisa menggunakan klausa ***using*** contoh:

```
select list_kolom
from t1
right join t2 using (kolom1);
```

Untuk join tabel lebih dari dua tabel, kita bisa menambahkan klausa ***left join*** yang ditunjukan dalam sintaks dibawah berikut:

```
select list_kolom
from t1
right join t2 on kondisi_join1
right join t3 on kondisi_join2
```

2.4.Full Join



Full Outer Join atau sering disingkat dengan Full Join akan mengembalikan seluruh baris dari kedua tabel yang dikenai ON termasuk data-data yang bernilai NULL, Tapi dalam DBMS MariaDB tidak disediakan sintaks bawaan untuk ***full join*** disamping ilustrasi diagramnya.

1.7 Manfaat join table dalam database SQL

Perintah JOIN dalam SQL digunakan untuk menampilkan data pada table yang saling berelasi atau tanpa berelasi tapi berhubungan. Artinya kita dapat menampilkan data dalam beberapa table dengan melihat ada kesamaan antar tabel walau ada entitas yang berbeda namun isinya dapat kita hubungkan.

1.8 Fungsi join table dalam database SQL

Pada bahasa SQL, operasi join atau penggabungan antar table adalah operasi dasar database relasional yang sangat penting. Untuk mendukung perancangan database relasional yang baik.

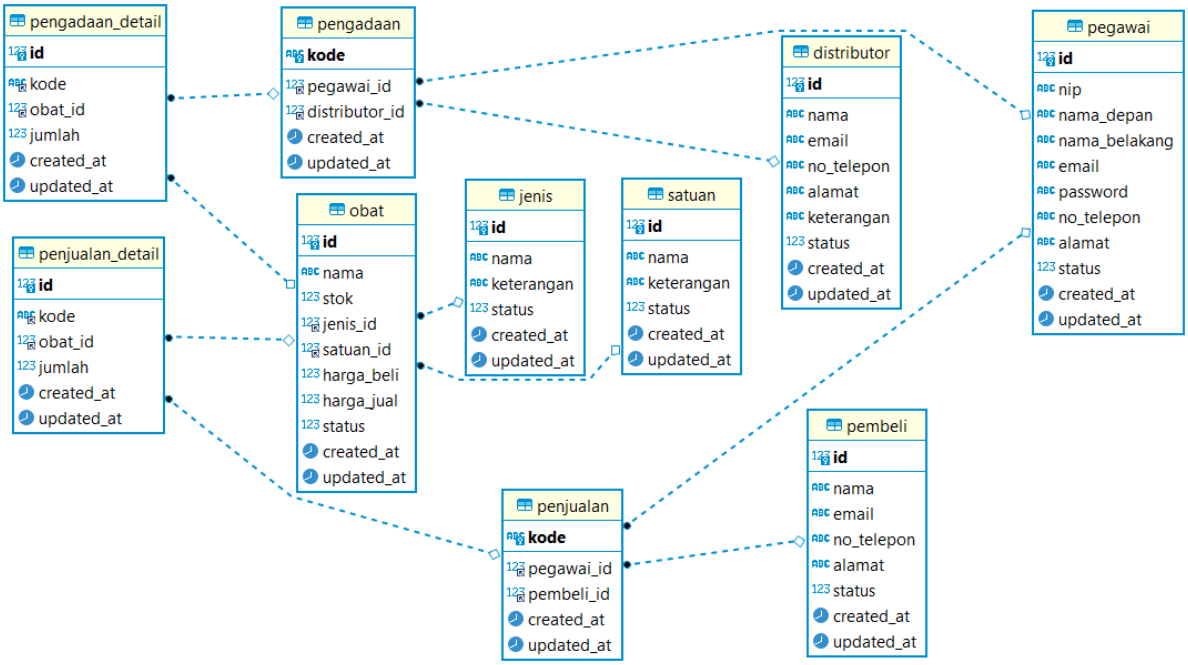
JOIN sangat penting untuk Anda pahami, jika ingin menghasilkan output data yang valid, menjamin integritas data, dan meminimalisir redudansi data.

BAB II STUDY KASUS

2.1 Deskripsi kasus yang diambil

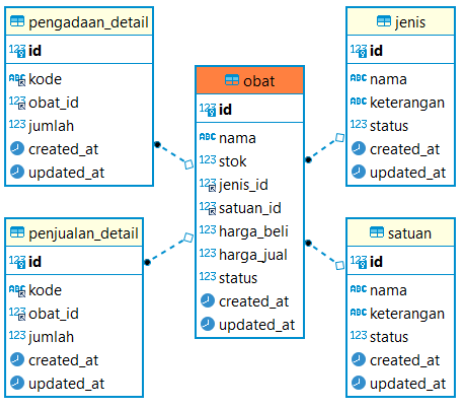
Pada makalah ini studi kasus bagaimana alur informasi obat masuk (Pengadaan) dan obat keluar (Penjualan) bisa kepada member (tabel pembeli) atau pembeli biasa. Ada beberapa kasus yang bisa diselesaikan dengan join dalam kasus ini.

2.2 Skema Database



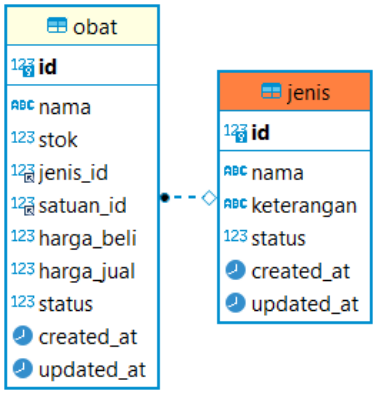
Dalam skema database diatas terdapat 10 tabel untuk mencatat transaksi obat masuk dan keluar penjelasan tabel nya dibawah sebagai berikut:

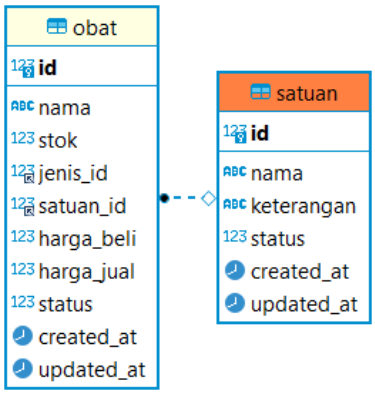
a) **Obat:** Tabel obat merupakan tabel utama dalam database ini karena transaksi di tabel ini untuk memantau obat yang masuk dan keluar. tabel obat berelasi ke beberapa tabel bisa dilihat dibawah:



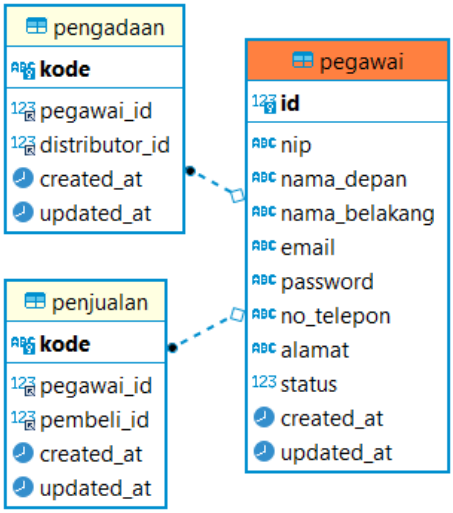
Dari gambar disamping diketahi bahwa tabel obat mempunyai relasi ke tabel ***jenis*** dan ***satuan*** yang kardinalitas nya ***many to one*** dimana satu ***jenis*** atau ***satuan*** bisa mempuyai banyak ***obat***. Kemudian kardinalitas antara ***pengadaan_detail*** dan ***penjualan_detail*** dengan obat yaitu ***one to one*** satu ke satu.

b) **Jenis dan satuan**: kedua tabel ini digunakan sebagai pelengkap tabel obat yang dimana relasi kardinalitas nya sudah dijelaskan diatas. Dibawah diagram tabel ***jenis*** dan ***satuan***:

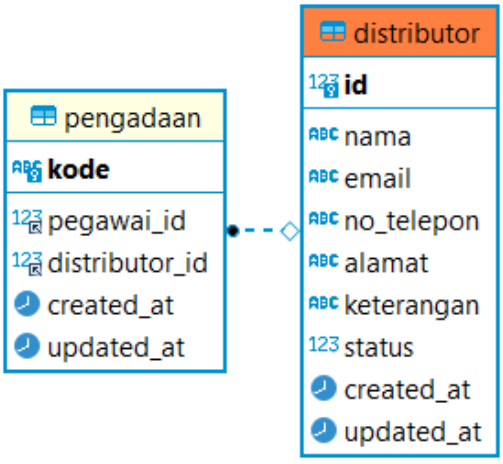




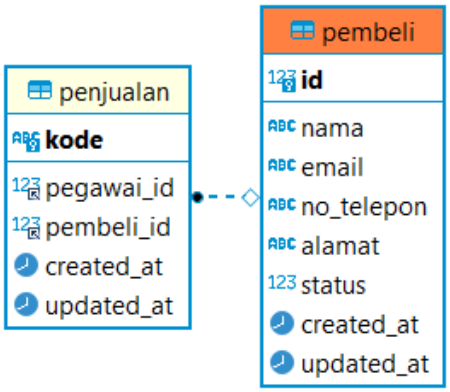
c) **Pegawai**: tabel ***pegawai*** digunakan untuk menyimpan data pegawai untuk setiap transaksi baik pengadaan dan penjualan. Tabel ***pegawai*** mempunyai relasi ke tabel ***pengadaan*** dan ***penjualan*** yang dimana kardinalitas relasi antara ***pegawai*** dengan ***pengadaan*** dan ***penjualan yaitu one to many***, jadi setiap ***pegawai*** bisa mempuyai banyak transaksi ***pengadaan*** maupun ***penjualan***, Untuk diagram bisa dilihat di gambar disamping.



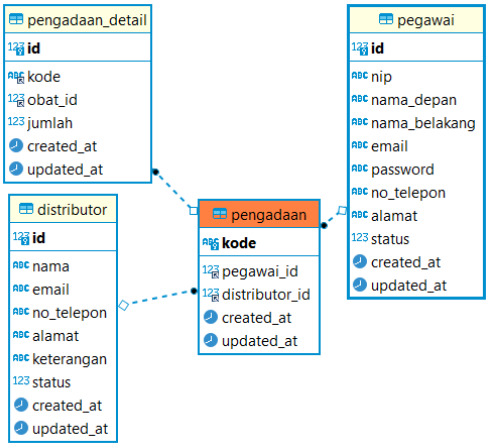
d) **Distributor:** Tabel ***distributor*** digunakan untuk ***pengadaan*** obat ***yang*** dimana setiap ***pengadaan*** harus ada sumber ***obat*** tersebut yaitu dari ***distributor***. Tabel ***distributor*** mempuyai relasi ke ***pengadaan*** yang dimana kardinalitas nya ***one to many*** dimana satu distributor bisa mempuyai (melakukan) banyak ***pengadaan***. Untuk diagram bisa dilihat di gambar disamping.



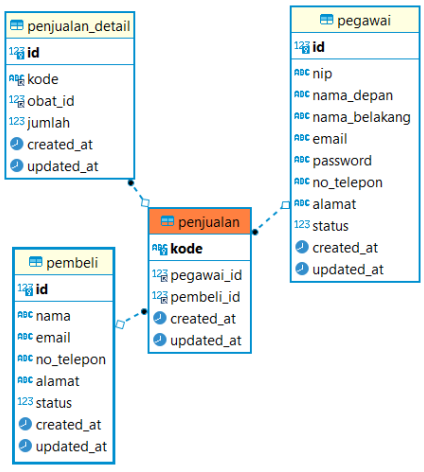
e) **Pembeli:** Tabel ***pembeli*** di diibaratkan sebagai member di apotek ini digunakan saat transaksi ***penjualan***. Tapi saat transaksi ***penjualan*** bisa juga tidak merelasikan tabel ***pembeli*** jika pembeli nya bukan member dari apotek ini dengan mengisi ***null***. Tabel ini berelasi ke tabel ***penjualan*** dimana relasi kardinalitas nya ***one to many*** dimana satu ***pembeli*** bisa mempuyai banyak ***penjualan*** (bisa membeli berkali kali). Untuk diagram bisa dilihat di gambar disamping.



f) **Pengadaan dan pengadaan_detail:** Tabel pengadaan digunakan untuk menyimpan data stok obat masuk atau pengadaan obat yang dilakukan oleh pegawai. Tabel ini mempunyai relasi ke beberapa tabel diantaranya ke tabel pengawai untu mencatat pegawai yang melakukan pengadaan dan pengadaan_detail untuk list data obat untuk pengadaan tersebut. untuk kardinalitas distributor dan pegawai terhadap penjualan sudah di jelaskan diatas dan untuk relasi kardinalitas antara tabel pengadaan dan pengadaan_detail yaitu one to many jadi setiap satu pengadaan mempunyai banyak detail untuk menampung beragam obat dalam pengadaan tersebut. Untuk diagram bisa dilihat di gambar disamping.



g) **Penjualan dan penjualan_detail:** hampir sama seperti pengadaan, pada penjualan juga mempunyai tabel detail nya yaitu penjualan_detail untuk menampung list stok obat yang keluar atau penjualan. Tabel ini juga berelasi ke tabel pegawai untuk mengetahui siapa yang melakukan penjualan dan pembeli penjualan kepada siapa. Namun untuk pembeli bisa di kosongkan karena tidak mungkin semua pembeli adalah member atau orang yang



tercatat dalam tabel pembeli. Tabel penjualan berelasi ke penjualan detail dengan kardinalitas one to many dimana satu penjualan bisa mempunyai banyak detail. Untuk diagram bisa dilihat di gambar disamping atas.

2.3 SQL Data Definition Language

```
START TRANSACTION;
--
-- Table structure for table `distributor`
--

CREATE TABLE `distributor` (
  `id` int(11) NOT NULL,
  `nama` varchar(255) DEFAULT NULL,
  `email` varchar(255) DEFAULT NULL,
  `no_telepon` varchar(255) DEFAULT NULL,
  `alamat` text DEFAULT NULL,
  `keterangan` text DEFAULT NULL,
  `status` int(11) DEFAULT NULL COMMENT '0 Tidak Aktif | 1 Aktif',
  `created_at` datetime NOT NULL DEFAULT current_timestamp(),
  `updated_at` datetime NOT NULL DEFAULT current_timestamp() ON UPDATE
current_timestamp(),
  PRIMARY KEY (`id`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4;

-- --------------------------------------------------------

--
-- Table structure for table `jenis`
--

CREATE TABLE `jenis` (
  `id` int(11) NOT NULL,
  `nama` varchar(255) DEFAULT NULL,
  `keterangan` text DEFAULT NULL,
  `status` int(11) DEFAULT NULL COMMENT '0 Tidak Aktif | 1 Aktif',
  `created_at` datetime NOT NULL DEFAULT current_timestamp(),
  `updated_at` datetime NOT NULL DEFAULT current_timestamp() ON UPDATE
current_timestamp(),
  PRIMARY KEY (`id`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4;

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

```sql
--
-- Table structure for table `obat`
--

CREATE TABLE `obat` (
  `id` int(11) NOT NULL,
  `nama` varchar(255) DEFAULT NULL,
  `stok` int(11) DEFAULT NULL,
  `jenis_id` int(11) DEFAULT NULL,
  `satuan_id` int(11) DEFAULT NULL,
  `harga_beli` bigint(20) DEFAULT NULL,
  `harga_jual` bigint(20) DEFAULT NULL,
  `status` int(11) DEFAULT NULL COMMENT '0 Tidak Aktif | 1 Aktif',
  `created_at` datetime NOT NULL DEFAULT current_timestamp(),
  `updated_at` datetime NOT NULL DEFAULT current_timestamp() ON UPDATE
current_timestamp(),
  PRIMARY KEY (`id`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4;

-- --------------------------------------------------------

--
-- Table structure for table `pegawai`
--

CREATE TABLE `pegawai` (
  `id` int(11) NOT NULL,
  `nip` varchar(255) DEFAULT NULL,
  `nama_depan` varchar(255) DEFAULT NULL,
  `nama_belakang` varchar(255) DEFAULT NULL,
  `email` varchar(255) DEFAULT NULL,
  `password` varchar(255) DEFAULT NULL,
  `no_telepon` varchar(255) DEFAULT NULL,
  `alamat` text DEFAULT NULL,
  `status` int(11) DEFAULT NULL COMMENT '0 Tidak Aktif | 1 Aktif',
  `created_at` datetime NOT NULL DEFAULT current_timestamp(),
  `updated_at` datetime NOT NULL DEFAULT current_timestamp() ON UPDATE
current_timestamp(),
  PRIMARY KEY (`id`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4;

-- --------------------------------------------------------

--
-- Table structure for table `pembeli`
--

CREATE TABLE `pembeli` (
  `id` int(11) NOT NULL,
  `nama` varchar(255) DEFAULT NULL,
  `email` varchar(255) DEFAULT NULL,
  `no_telepon` varchar(255) DEFAULT NULL,
  `alamat` text DEFAULT NULL,
  `status` int(11) DEFAULT NULL COMMENT '0 Tidak Aktif | 1 Aktif',
  `created_at` datetime NOT NULL DEFAULT current_timestamp(),
  `updated_at` datetime NOT NULL DEFAULT current_timestamp() ON UPDATE
current_timestamp(),
  PRIMARY KEY (`id`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4;
```

```sql
--
-- Table structure for table `pengadaan`
--

CREATE TABLE `pengadaan` (
  `kode` varchar(255) NOT NULL,
  `pegawai_id` int(11) DEFAULT NULL,
  `distributor_id` int(11) DEFAULT NULL,
  `created_at` datetime NOT NULL DEFAULT current_timestamp(),
  `updated_at` datetime NOT NULL DEFAULT current_timestamp() ON UPDATE
current_timestamp(),
  PRIMARY KEY (`kode`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4;

-- --------------------------------------------------------

--
-- Table structure for table `pengadaan_detail`
--

CREATE TABLE `pengadaan_detail` (
  `id` int(11) NOT NULL,
  `kode` varchar(255) DEFAULT NULL,
  `obat_id` int(11) DEFAULT NULL,
  `jumlah` int(11) DEFAULT NULL,
  `created_at` datetime NOT NULL DEFAULT current_timestamp(),
  `updated_at` datetime NOT NULL DEFAULT current_timestamp() ON UPDATE
current_timestamp(),
  PRIMARY KEY (`id`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4;

-- --------------------------------------------------------

--
-- Table structure for table `penjualan`
--

CREATE TABLE `penjualan` (
  `kode` varchar(255) NOT NULL,
  `pegawai_id` int(11) DEFAULT NULL,
  `pembeli_id` int(11) DEFAULT NULL,
  `created_at` datetime NOT NULL DEFAULT current_timestamp(),
  `updated_at` datetime NOT NULL DEFAULT current_timestamp() ON UPDATE
current_timestamp(),
  PRIMARY KEY (`kode`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4;

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

```sql
--
-- Table structure for table `penjualan_detail`
--

CREATE TABLE `penjualan_detail` (
  `id` int(11) NOT NULL,
  `kode` varchar(255) DEFAULT NULL,
  `obat_id` int(11) DEFAULT NULL,
  `jumlah` int(11) DEFAULT NULL,
  `created_at` datetime NOT NULL DEFAULT current_timestamp(),
  `updated_at` datetime NOT NULL DEFAULT current_timestamp() ON UPDATE
current_timestamp(),
  PRIMARY KEY (`id`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4;

-- --------------------------------------------------------

--
-- Table structure for table `satuan`
--

CREATE TABLE `satuan` (
  `id` int(11) NOT NULL,
  `nama` varchar(255) DEFAULT NULL,
  `keterangan` text DEFAULT NULL,
  `status` int(11) DEFAULT NULL COMMENT '0 Tidak Aktif | 1 Aktif',
  `created_at` datetime NOT NULL DEFAULT current_timestamp(),
  `updated_at` datetime NOT NULL DEFAULT current_timestamp() ON UPDATE
current_timestamp(),
  PRIMARY KEY (`id`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4;

--
-- Indexes for dumped tables
--

--
-- Indexes for table `obat`
--
ALTER TABLE `obat`
  ADD KEY `jenis_id` (`jenis_id`),
  ADD KEY `satuan_id` (`satuan_id`);

--
-- Indexes for table `pengadaan`
--
ALTER TABLE `pengadaan`
  ADD KEY `distributor_id` (`distributor_id`),
  ADD KEY `pegawai_id` (`pegawai_id`);

--
-- Indexes for table `pengadaan_detail`
--
ALTER TABLE `pengadaan_detail`
  ADD KEY `pengadaan_kode` (`kode`),
  ADD KEY `obat_id` (`obat_id`);

--
```

```
--
-- Indexes for table `penjualan`
--
ALTER TABLE `penjualan`
  ADD KEY `pembeli_id` (`pembeli_id`),
  ADD KEY `pegawai_id` (`pegawai_id`);

--
-- Indexes for table `penjualan_detail`
--
ALTER TABLE `penjualan_detail`
  ADD KEY `penjualan_kode` (`kode`),
  ADD KEY `obat_id` (`obat_id`);

--
-- Constraints for table `obat`
--
ALTER TABLE `obat`
  ADD CONSTRAINT `obat_ibfk_1` FOREIGN KEY (`jenis_id`) REFERENCES
`jenis` (`id`),
  ADD CONSTRAINT `obat_ibfk_2` FOREIGN KEY (`satuan_id`) REFERENCES
`satuan` (`id`);

--
-- Constraints for table `pengadaan`
--
ALTER TABLE `pengadaan`
  ADD CONSTRAINT `pengadaan_ibfk_1` FOREIGN KEY (`distributor_id`)
REFERENCES `distributor` (`id`),
  ADD CONSTRAINT `pengadaan_ibfk_2` FOREIGN KEY (`pegawai_id`) REFERENCES
`pegawai` (`id`);

--
-- Constraints for table `pengadaan_detail`
--
ALTER TABLE `pengadaan_detail`
  ADD CONSTRAINT `pengadaan_detail_ibfk_1` FOREIGN KEY (`kode`)
REFERENCES `pengadaan` (`kode`),
  ADD CONSTRAINT `pengadaan_detail_ibfk_2` FOREIGN KEY (`obat_id`)
REFERENCES `obat` (`id`);

--
-- Constraints for table `penjualan`
--
ALTER TABLE `penjualan`
  ADD CONSTRAINT `penjualan_ibfk_1` FOREIGN KEY (`pembeli_id`) REFERENCES
`pembeli` (`id`),
  ADD CONSTRAINT `penjualan_ibfk_2` FOREIGN KEY (`pegawai_id`) REFERENCES
`pegawai` (`id`);

--
-- Constraints for table `penjualan_detail`
--
ALTER TABLE `penjualan_detail`
  ADD CONSTRAINT `penjualan_detail_ibfk_1` FOREIGN KEY (`kode`)
REFERENCES `penjualan` (`kode`),
  ADD CONSTRAINT `penjualan_detail_ibfk_2` FOREIGN KEY (`obat_id`)
REFERENCES `obat` (`id`);
COMMIT;
```

2.4 SQL Data Manipulation Language

```
START TRANSACTION;
--
-- Dumping data for table `jenis`
--

INSERT INTO `jenis` (`id`, `nama`, `keterangan`, `status`, `created_at`,
`updated_at`) VALUES
(1, 'Cair', 'ket', 1, '2021-10-07 12:31:45', '2021-10-07 12:31:45'),
(2, 'Tablet', 'ket', 1, '2021-10-07 12:31:45', '2021-10-07 12:31:45'),
(3, 'Kapsul', 'ket', 1, '2021-10-07 12:31:45', '2021-10-07 12:31:45'),
(4, 'Pill', 'ket', 1, '2021-10-07 12:31:45', '2021-10-07 12:31:45'),
(5, 'Serbuk', 'ket', 1, '2021-10-07 12:31:45', '2021-10-07 12:31:45'),
(6, 'Salep', 'ket', 1, '2021-10-07 12:31:45', '2021-10-07 12:31:45'),
(7, 'Tetes', 'ket', 1, '2021-10-07 12:31:45', '2021-10-07 12:31:45');

--
-- Dumping data for table `satuan`
--

INSERT INTO `satuan` (`id`, `nama`, `keterangan`, `status`,
`created_at`, `updated_at`) VALUES
(1, 'Box', NULL, 1, '2021-10-07 12:33:24', '2021-10-07 12:33:24'),
(2, 'Pcs', NULL, 1, '2021-10-07 12:33:24', '2021-10-07 12:33:34');

--
-- Dumping data for table `obat`
--

INSERT INTO `obat` (`id`, `nama`, `stok`, `jenis_id`, `satuan_id`,
`harga_beli`, `harga_jual`, `status`, `created_at`, `updated_at`) VALUES
(1, 'Paracetamol', 1, 2, 2, 2500, 5000, 1, '2021-10-07 12:34:48', '2021-
10-09 16:37:54'),
(2, 'Kataflam', 0, 2, 2, 10000, 15000, 1, '2021-10-07 12:41:44', '2021-
10-07 12:41:44'),
(3, 'Antimo', 1, 1, 2, 5000, 7500, 1, '2021-10-07 12:41:44', '2021-10-09
23:35:03'),
(4, 'Tramadol', 0, 1, 1, 10000, 15000, 1, '2021-10-07 12:49:07', '2021-
10-09 16:38:31'),
(5, 'PIROXICAM', 0, 2, 1, 100000, 150000, 1, '2021-10-09 13:38:36',
'2021-10-09 16:38:27'),
(6, 'Hydrocorti', 1, 6, 2, 10000, 13000, 1, '2021-10-09 13:38:36',
'2021-10-09 16:38:20'),
(7, 'Acyclovir Cream', 1, 6, 2, 7000, 10000, 1, '2021-10-09 13:40:26',
'2021-10-09 16:38:17'),
(8, 'Ethambutol', 2, 2, 2, 20000, 21000, 1, '2021-10-09 13:40:26',
'2021-10-09 16:38:13'),
(9, 'Promag', 5, 2, 1, 13000, 15000, 1, '2021-10-09 13:43:41', '2021-10-
09 16:38:10'),
(10, 'Nebacetin', 4, 5, 2, 20000, 25000, 1, '2021-10-09 13:43:41',
'2021-10-09 16:38:05');
```

```sql
--
-- Dumping data for table `pegawai`
--

INSERT INTO `pegawai` (`id`, `nip`, `nama_depan`, `nama_belakang`,
`email`, `password`, `no_telepon`, `alamat`, `status`, `created_at`,
`updated_at`) VALUES
(1, '2113191234', 'Isep Lutpi', 'Nur', 'iseplutfinur@gmail.com',
'isep1234', '0888134689793', 'Sekeloa, Bandung', 1, '2021-10-09
13:49:29', '2021-10-09 13:59:29'),
(2, '2113174658', 'Adistia ', 'Ramadhani', 'adistiaramadhani@gmail.com',
'adistia123', '089364765685', 'Antapani,Bandung', 1, '2021-10-09
13:49:29', '2021-10-09 13:49:29'),
(3, '21131746568', 'Alam', 'Nurzaman', 'alamnurzaman@gmail.com',
'alam1234', '088834561239', 'Ciumbuleuit, Bandung', 1, '2021-10-09
13:55:47', '2021-10-09 13:55:47'),
(4, '2113191098', 'Dara Atria', 'Ferliandini',
'daraatriaferliandini@gmail.com', 'dara1234', '0889391328', 'Cikutra
Barat, Bandung', 1, '2021-10-09 13:55:47', '2021-10-09 13:55:47');


--
-- Dumping data for table `distributor`
--

INSERT INTO `distributor` (`id`, `nama`, `email`, `no_telepon`,
`alamat`, `keterangan`, `status`, `created_at`, `updated_at`) VALUES
(1, 'Pt Kimia farma', 'kimiafarma@gmail.com', '085798132505', 'Bandung',
'Dumy', 1, '2021-10-07 12:28:40', '2021-10-07 12:28:40'),
(2, 'Pt obat sakti', 'sakti', '123', '123', 'ket', 1, '2021-10-07
12:28:40', '2021-10-07 12:28:40');

--
-- Dumping data for table `pembeli`
--

INSERT INTO `pembeli` (`id`, `nama`, `email`, `no_telepon`, `alamat`,
`status`, `created_at`, `updated_at`) VALUES
(1, 'Zayn Malik', 'zaynmalik@gmail.com', '08889391329', 'Bojongkoneng',
1, '2021-10-09 13:58:45', '2021-10-09 13:58:45'),
(2, 'Harry Styles', 'harrys@gmail.com', '088854328586', 'Cibangkong', 1,
'2021-10-09 13:58:45', '2021-10-09 13:58:45'),
(3, 'Gigi Hadid', 'gigihadid@gmail.com', '088938656877', 'Sadang
Serang', 1, '2021-10-09 14:01:49', '2021-10-09 14:01:49'),
(4, 'Kendall Jenner', 'kendaljnn@gmail.com', '089374656836', 'Cicaheum',
1, '2021-10-09 14:01:49', '2021-10-09 14:01:49');
```

```sql
--
-- Dumping data for table `pengadaan`
--

INSERT INTO `pengadaan` (`kode`, `pegawai_id`, `distributor_id`,
`created_at`, `updated_at`) VALUES
('SM-2021011', 1, 1, '2021-01-01 14:16:28', '2021-10-09 23:36:11'),
('SM-2021012', 1, 1, '2021-01-15 14:16:28', '2021-10-09 23:36:13'),
('SM-2021021', 2, 2, '2021-02-01 14:16:28', '2021-10-09 23:36:17'),
('SM-2021022', 1, 1, '2021-02-15 14:16:28', '2021-10-09 23:36:15'),
('SM-2021031', 1, 2, '2021-03-01 14:16:28', '2021-10-09 23:36:21'),
('SM-2021032', 3, 1, '2021-03-15 14:16:28', '2021-10-09 23:36:27'),
('SM-2021041', 1, 1, '2021-04-01 14:16:28', '2021-10-09 23:36:34'),
('SM-2021042', 1, 2, '2021-04-15 14:16:28', '2021-10-09 23:36:31'),
('SM-2021051', 1, 1, '2021-05-01 14:16:28', '2021-10-09 23:36:29'),
('SM-2021052', 3, 1, '2021-05-15 14:16:28', '2021-10-09 23:36:38');

--
-- Dumping data for table `pengadaan_detail`
--

INSERT INTO `pengadaan_detail` (`id`, `kode`, `obat_id`, `jumlah`,
`created_at`, `updated_at`) VALUES
(1, 'SM-2021011', 1, 5, '2021-01-01 14:44:20', '2021-01-01 14:44:20'),
(2, 'SM-2021012', 2, 3, '2021-01-15 14:44:20', '2021-01-15 14:44:20'),
(3, 'SM-2021021', 3, 2, '2021-02-01 14:45:44', '2021-02-01 14:45:44'),
(4, 'SM-2021022', 4, 5, '2021-02-15 14:45:44', '2021-02-15 14:45:44'),
(5, 'SM-2021031', 5, 7, '2021-03-01 14:47:15', '2021-03-01 14:47:15'),
(6, 'SM-2021032', 6, 5, '2021-03-15 14:47:15', '2021-03-15 14:47:15'),
(7, 'SM-2021041', 7, 4, '2021-04-01 14:52:11', '2021-04-01 14:52:11'),
(8, 'SM-2021042', 8, 8, '2021-04-15 14:52:11', '2021-04-15 14:52:11'),
(9, 'SM-2021051', 9, 5, '2021-05-01 14:55:55', '2021-05-01 14:55:55'),
(10, 'SM-2021052', 3, 1, '2021-05-15 14:55:55', '2021-10-09 23:35:21');

--
-- Dumping data for table `penjualan`
--

INSERT INTO `penjualan` (`kode`, `pegawai_id`, `pembeli_id`,
`created_at`, `updated_at`) VALUES
('SK-2021011', 4, NULL, '2021-01-06 15:00:34', '2021-10-09 16:03:13'),
('SK-2021012', 2, NULL, '2021-01-08 15:00:34', '2021-10-09 16:03:07'),
('SK-2021021', 1, 4, '2021-02-04 15:00:34', '2021-10-09 15:50:53'),
('SK-2021022', 3, NULL, '2021-02-06 15:00:34', '2021-10-09 15:52:30'),
('SK-2021031', 1, NULL, '2021-03-07 15:00:34', '2021-10-09 16:03:19'),
('SK-2021032', 2, 2, '2021-03-15 15:00:34', '2021-10-09 15:51:19'),
('SK-2021041', 3, 4, '2021-04-09 15:00:34', '2021-10-09 15:51:27'),
('SK-2021042', 2, 1, '2021-04-20 15:00:34', '2021-10-09 15:51:35'),
('SK-2021051', 1, NULL, '2021-05-13 15:00:34', '2021-10-09 15:52:40'),
('SK-2021052', 4, 2, '2021-05-16 15:00:34', '2021-10-09 15:51:55');
```

```sql
--
-- Dumping data for table `penjualan_detail`
--

INSERT INTO `penjualan_detail` (`id`, `kode`, `obat_id`, `jumlah`,
`created_at`, `updated_at`) VALUES
(11, 'SK-2021011', 1, 1, '2021-01-06 00:00:00', '2021-01-06 00:00:00'),
(12, 'SK-2021011', 3, 1, '2021-01-06 00:00:00', '2021-01-06 00:00:00'),
(13, 'SK-2021011', 4, 3, '2021-01-06 00:00:00', '2021-01-06 00:00:00'),
(14, 'SK-2021012', 1, 2, '2021-08-01 00:00:00', '2021-08-01 00:00:00'),
(15, 'SK-2021012', 2, 1, '2021-08-01 00:00:00', '2021-08-01 00:00:00'),
(16, 'SK-2021021', 1, 1, '2021-02-04 00:00:00', '2021-02-04 00:00:00'),
(17, 'SK-2021021', 2, 1, '2021-02-04 00:00:00', '2021-02-04 00:00:00'),
(18, 'SK-2021021', 3, 1, '2021-02-04 00:00:00', '2021-02-04 00:00:00'),
(19, 'SK-2021021', 4, 2, '2021-02-04 00:00:00', '2021-02-04 00:00:00'),
(20, 'SK-2021022', 5, 1, '2021-02-06 00:00:00', '2021-02-06 00:00:00'),
(21, 'SK-2021022', 6, 1, '2021-02-06 00:00:00', '2021-02-06 00:00:00'),
(22, 'SK-2021031', 5, 1, '2021-03-07 00:00:00', '2021-03-07 00:00:00'),
(23, 'SK-2021031', 6, 1, '2021-03-07 00:00:00', '2021-03-07 00:00:00'),
(24, 'SK-2021032', 5, 1, '0000-00-00 00:00:00', '0000-00-00 00:00:00'),
(25, 'SK-2021032', 6, 1, '0000-00-00 00:00:00', '0000-00-00 00:00:00'),
(26, 'SK-2021032', 7, 3, '0000-00-00 00:00:00', '0000-00-00 00:00:00'),
(27, 'SK-2021032', 8, 2, '0000-00-00 00:00:00', '0000-00-00 00:00:00'),
(28, 'SK-2021041', 5, 1, '2021-09-04 00:00:00', '2021-09-04 00:00:00'),
(29, 'SK-2021041', 6, 1, '2021-09-04 00:00:00', '2021-09-04 00:00:00'),
(30, 'SK-2021042', 5, 1, '0000-00-00 00:00:00', '0000-00-00 00:00:00'),
(31, 'SK-2021042', 8, 2, '0000-00-00 00:00:00', '0000-00-00 00:00:00'),
(32, 'SK-2021051', 5, 1, '0000-00-00 00:00:00', '0000-00-00 00:00:00'),
(33, 'SK-2021051', 8, 2, '0000-00-00 00:00:00', '0000-00-00 00:00:00'),
(34, 'SK-2021052', 5, 1, '0000-00-00 00:00:00', '0000-00-00 00:00:00');

COMMIT;
```

BAB III PEMBAHASAN STUDY KASUS

3.1 Mengambil data obat saat ini menggunakan inner join

```
select obat.*, jenis.nama as nama_jenis, satuan.nama as nama_satuan
from obat
inner join jenis on obat.jenis_id = jenis.id
inner join satuan on obat.satuan_id = satuan.id
order by obat.id
```

	id	nama	stok	jenis_id	satuan_id	harga_beli	harga_jual	status	created_at	updated_at	nama_jenis	nama_satuan
1	1	Paracetamol	1	2	2	2,500	5,000	1	2021-10-07 12:34:48	2021-10-09 16:37:54	Tablet	Pcs
2	2	Kataflam	0	2	2	10,000	15,000	1	2021-10-07 12:41:44	2021-10-07 12:41:44	Tablet	Pcs
3	3	Antimo	1	1	2	5,000	7,500	1	2021-10-07 12:41:44	2021-10-09 23:35:03	Cair	Pcs
4	4	Tramadol	0	1	1	10,000	15,000	1	2021-10-07 12:49:07	2021-10-09 16:38:31	Cair	Box
5	5	PIROXICAM	0	2	1	100,000	150,000	1	2021-10-09 13:38:36	2021-10-09 16:38:27	Tablet	Box
6	6	Hydrocorti	1	6	2	10,000	13,000	1	2021-10-09 13:38:36	2021-10-09 16:38:20	Salep	Pcs
7	7	Acyclovir Cream	1	6	2	7,000	10,000	1	2021-10-09 13:40:26	2021-10-09 16:38:17	Salep	Pcs
8	8	Ethambutol	2	2	2	20,000	21,000	1	2021-10-09 13:40:26	2021-10-09 16:38:13	Tablet	Pcs
9	9	Promag	5	2	1	13,000	15,000	1	2021-10-09 13:43:41	2021-10-09 16:38:10	Tablet	Box
10	10	Nebacetin	4	5	2	20,000	25,000	1	2021-10-09 13:43:41	2021-10-09 16:38:05	Serbuk	Pcs

3.2 Mengambil data obat dengan jenis kapsul(id:3) menggunakan right join

```
select obat.*, jenis.nama as nama_jenis
from obat
right join jenis on obat.jenis_id = jenis.id
where jenis.id = 3
order by obat.id
```

	id	nama	stok	jenis_id	satuan_id	harga_beli	harga_jual	status	created_at	updated_at	nama_jenis
1	[NULL]	[NULL]	[NULL]	[NULL]	[NULL]	[NULL]	[NULL]	[NULL]	[NULL]	[NULL]	Kapsul

3.3 Mengambil semua jenis obat menggunakan left join

```
select obat.nama as nama_obat, jenis.nama as nama_jenis
from jenis
left join obat on obat.jenis_id = jenis.id
order by obat.id
```

	nama_obat	nama_jenis
1	[NULL]	Kapsul
2	[NULL]	Pill
3	[NULL]	Tetes
4	Paracetamol	Tablet
5	Kataflam	Tablet
6	Antimo	Cair
7	Tramadol	Cair
8	PIROXICAM	Tablet
9	Hydrocorti	Salep
10	Acyclovir Cream	Salep
11	Ethambutol	Tablet
12	Promag	Tablet
13	Nebacetin	Serbuk

BAB IV PENUTUP

4.1 Kesimpulan

Join adalah penggabungan table yang dilakukan melalui kolom / key tertentu yang memiliki nilai terkait untuk mendapatkan satu set data dengan informasi lengkap. Lengkap disini artinya kolom data didapatkan dari kolom-kolom hasil join antar table tersebut. Join diperlukan karena perancangan table pada sistem transaksional kebanyakan di-normalisasi, salah satu alasannya untuk menghindari redundansi data.

Pada bahasa SQL, operasi join atau penggabungan antar table adalah operasi dasar database relasional yang sangat penting. Untuk mendukung perancangan database relasional yang baik.

Perintah JOIN dalam SQL digunakan untuk menampilkan data pada table yang saling berelasi atau tanpa berelasi tapi berhubungan. Artinya kita dapat menampilkan data dalam beberapa table dengan melihat ada kesamaan antar tabel walau ada entitas yang berbeda namun isinya dapat kita hubungkan. Join table terbagi menjadi beberapa jenis yaitu inner join, left join, right join dan full join.

Semua database dan asset makalah ini disimpan dalam repositori github di url:
https://github.com/iseplutpinur/makalah_dbms_mariadb_join

DAFTAR PUSTAKA

1. Ccodepolitan. CodePolitan.com. Published 2021. Accessed October 9, 2021. https://www.codepolitan.com/tujuh-teknik-join-di-sql-596c537f0deb3
2. A Visual Explanation of MariaDB Joins with Practical Examples. MariaDB Tutorial. Published April 11, 2020. Accessed October 9, 2021. https://www.mariadbtutorial.com/mariadb-basics/mariadb-join/
3. MariaDB Inner Join. MariaDB Tutorial. Published April 11, 2020. Accessed October 9, 2021. https://www.mariadbtutorial.com/mariadb-basics/mariadb-inner-join/
4. MariaDB Left Join. MariaDB Tutorial. Published April 11, 2020. Accessed October 9, 2021. https://www.mariadbtutorial.com/mariadb-basics/mariadb-left-join/
5. M Fikri Setiadi. 3 Fungsi Join pada Mysql yang Wajib Anda Ketahui. mfikri. Published 2017. Accessed October 9, 2021. https://mfikri.com/artikel/3-fungsi-join-pada-mysql-yang-wajib-anda-ketahui.html