DATABASE SQL

Pesantren PeTIK II YBM PLN
Jl. KH. Bisri Syansuri RT/01 RW/05, Plosogeneng,
Kec. Jombang, Kabupaten Jombang, Jawa Timur

Pertemuan Ke-13

Materi

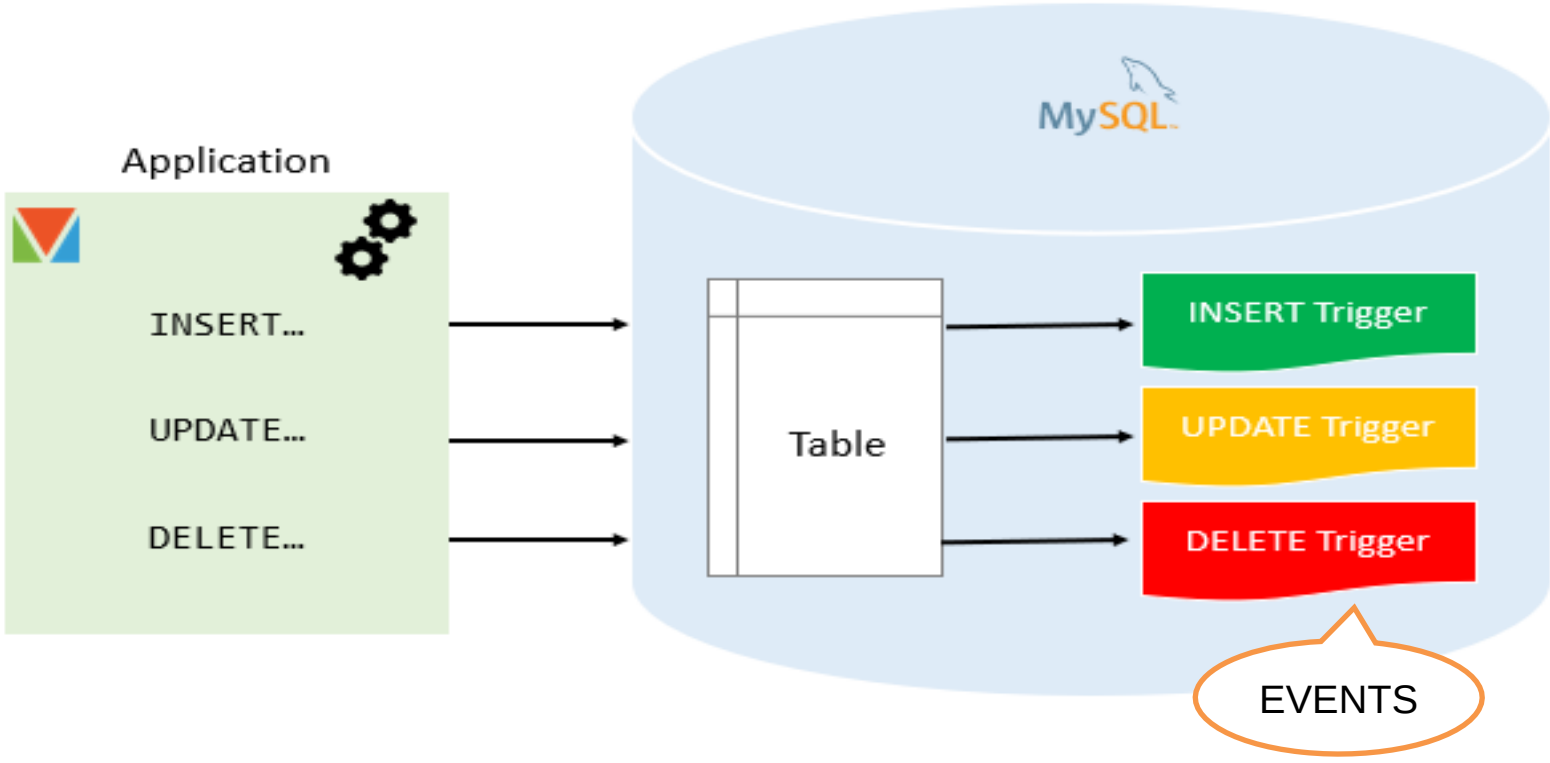
1. Pengantar Database
2. Pemodelan Data
3. Model Relasional Database
4. Normalisasi Database
5. Pengantar SQL
6. Perintah SQL SELECT 1
7. Perintah SQL SELECT 2
9. Fungsi Aggregate dan Grouping Data
10. Sub Query & SQL Join Table
11. View dan Analisa Query
12. Store Procedure dan Function
13. **Trigger dan Transaction**
14. Manajemen User
15. Backup dan Restore

13. Trigger dan Transaction

Triggers

- Triggers: Kode program SQL (Procedural) yang dijalankan secara otomatis di Server Database
- Trigger menanggapi proses DML pada suatu Table, proses DML:
  - INSERT
  - UPDATE
  - DELETE
- Terdapat event pada table untuk mengeksekusi triggers yaitu:
  - BEFORE : Sebelum eksekusi trigger
  - AFTER : Setelah eksekusi trigger

MySQL - Triggers



https://www.mysqltutorial.org/mysql-triggers.aspx/

Triggers

- Trigger adalah salah satu cara untuk memastikan aturan proses bisnis terjadi dan sekaligus melindungi integritas data
- Trigger dapat dikatakan sebagai store procedure spesial yang dijalankan pada suatu event (kejadian).
- Contohnya : trigger kurangi_stok akan dieksekusi ketika terjadi proses transaksi penjualan produk yang mengakibatkan stok pada tabel produk berkurang
- Melihat Trigger yang ada:

  mysql> show triggers;

Perbedaan :
Triggers & Store Procedure

- Trigger akan dieksekusi ketika merespon suatu event (kejadian), sedangkan store procedure tidak merespon event yang terjadi pada database
- Store Procedure dapat memiliki satu atau lebih inputan parameter. Sementara trigger tidak dapat menerima inputan paramater
- Sebuah Store Procedure dapat mempunyai nilai balik, sementara sebuah trigger tidak mempunyai nilai balik

Syntax Trigger

```
CREATE
[DEFINER = { user | CURRENT_USER }]
TRIGGER trigger_name
trigger_time trigger_event
ON tbl_name FOR EACH ROW
trigger_body
trigger_time: { BEFORE | AFTER }
trigger_event: { INSERT | UPDATE | DELETE }
```

Syntax Trigger

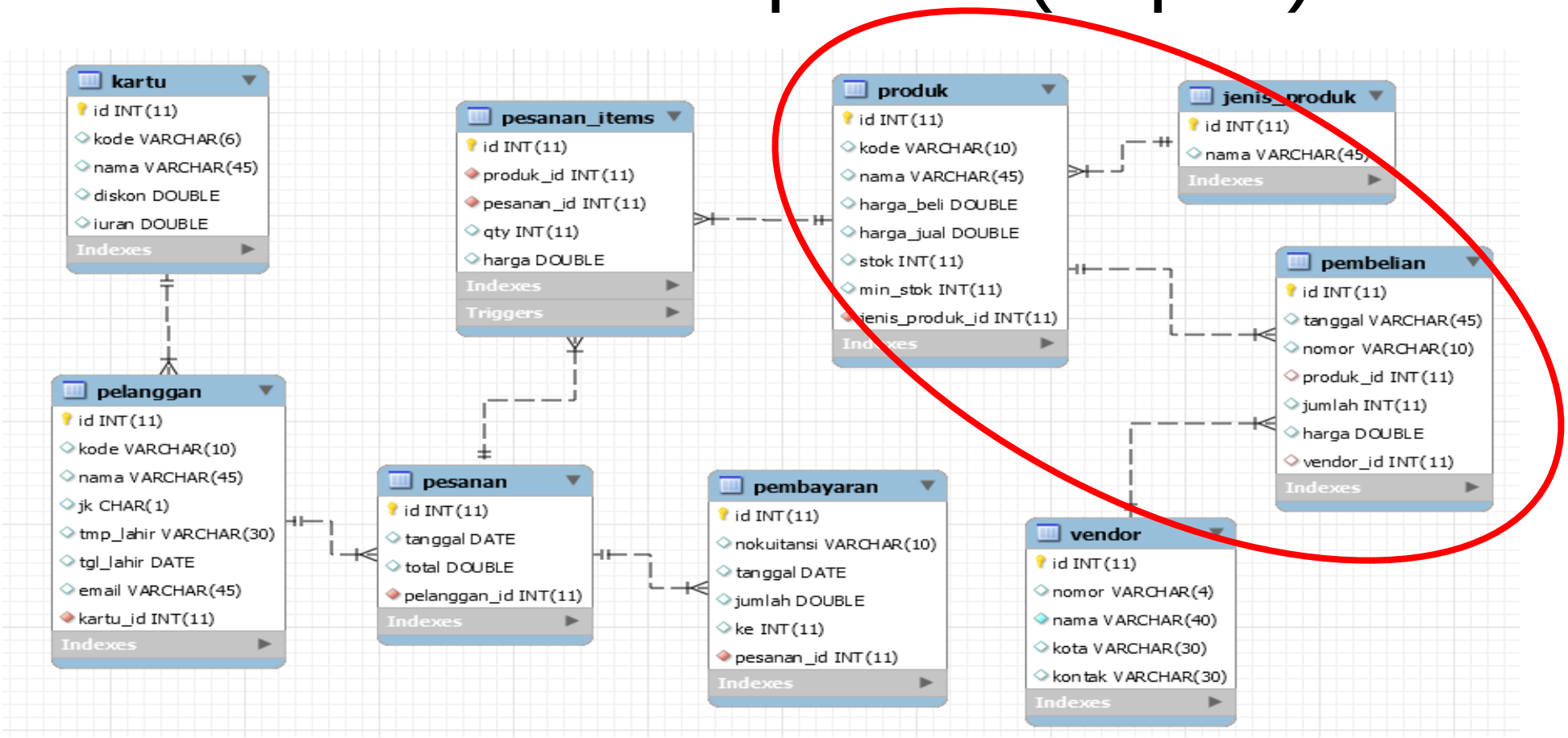
```
1  mysql> delimiter //
2  mysql> CREATE TRIGGER upd_check BEFORE UPDATE ON account
3         FOR EACH ROW
4         BEGIN
5             IF NEW.amount < 0 THEN
6                 SET NEW.amount = 0;
7             ELSEIF NEW.amount > 100 THEN
8                 SET NEW.amount = 100;
9             END IF;
10        END;//
11 mysql> delimiter ;
```

Contoh :: Trigger pembelian produk

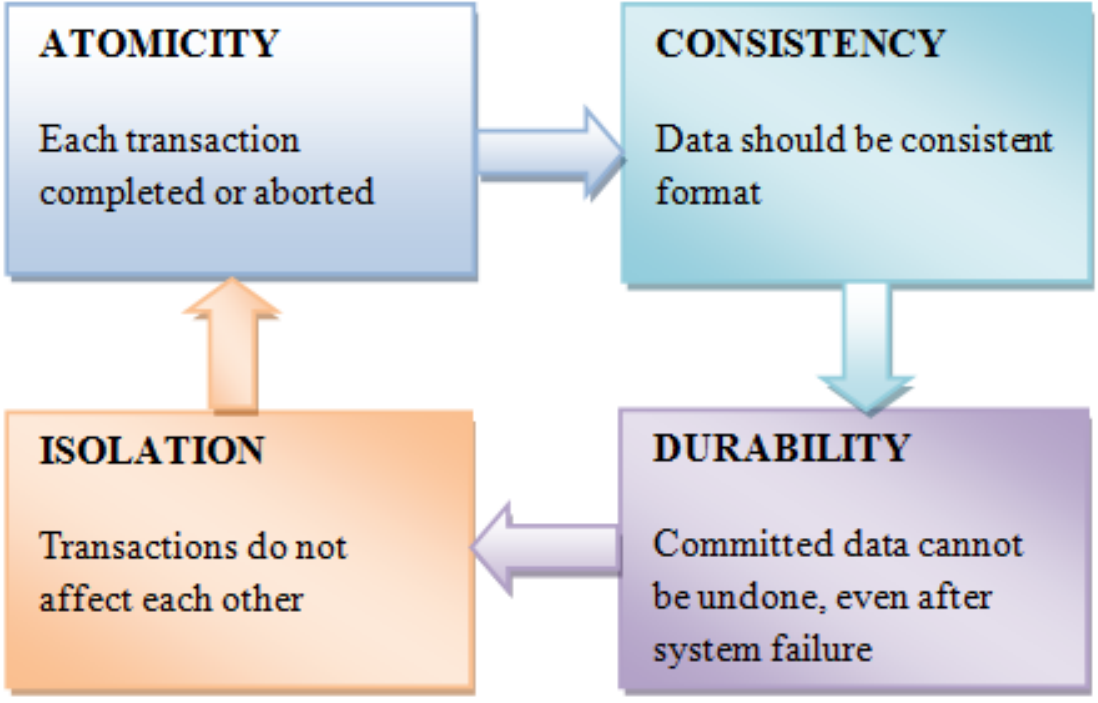
- Untuk membuat Trigger anda harus sebagai user root
- Berikut trigger yang melakukan penambahan stok saat transkasi pembelian terjadi pada table pembelian

```
mysql> delimiter //
mysql> CREATE TRIGGER trig_tambah_stok AFTER INSERT ON
    pembelian
     -> FOR EACH ROW BEGIN
     -> UPDATE produk SET stok = stok + NEW.jumlah WHERE
    id=NEW.produk_id;
     ->  END;
     -> //
mysql> delimiter ;
```

ERD Database Koperasi (dbpos)



Transaction Database : ACID



A = Atomicity

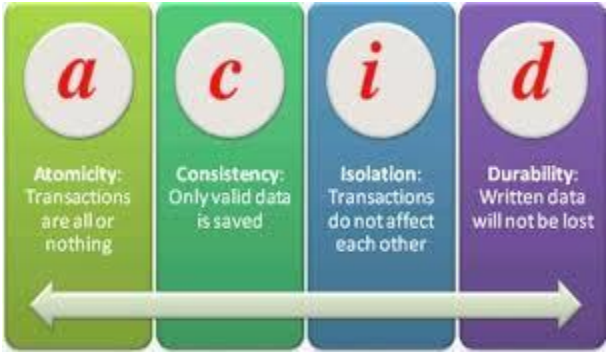
C = Consistency

I = Isolation

D = Durability

Transaction Database :: ACID

- **Atomicity**
  - Setiap transaksi sql bersifat atomic “semua diekskusi atau tidak sama sekali”
- **Consistency**
  - Memastikan semua transaksi bersifat konsisten dari satu state ke state lainnya



- **Isolation**
  - Sebuah transaksi bersifat independensi dan terisolasi, sebuah transaksi dipastikan tidak mempengaruhi transaksi lainnya
- **Durability**
  - Memastikan transaksi yang telah dilakukan (commited) tidak mengakibatkan hilanganya data

Transaction MySQL

- Proses Transaksi: melibatkan eksekusi kumpulan query yang dijalankan dalam satu blok perintah.
- Sebuah transaksi database harus memenuhi 4 property ACID
- **Commit** : Jika eksekusi query berhasil, dan commit diterapkan agar tersimpan dalam database
- **Rollback**: Jika ada satu kasus query gagal, maka proses query dibatalkan, data kembali ke kondisi semula

Transaction MySQL

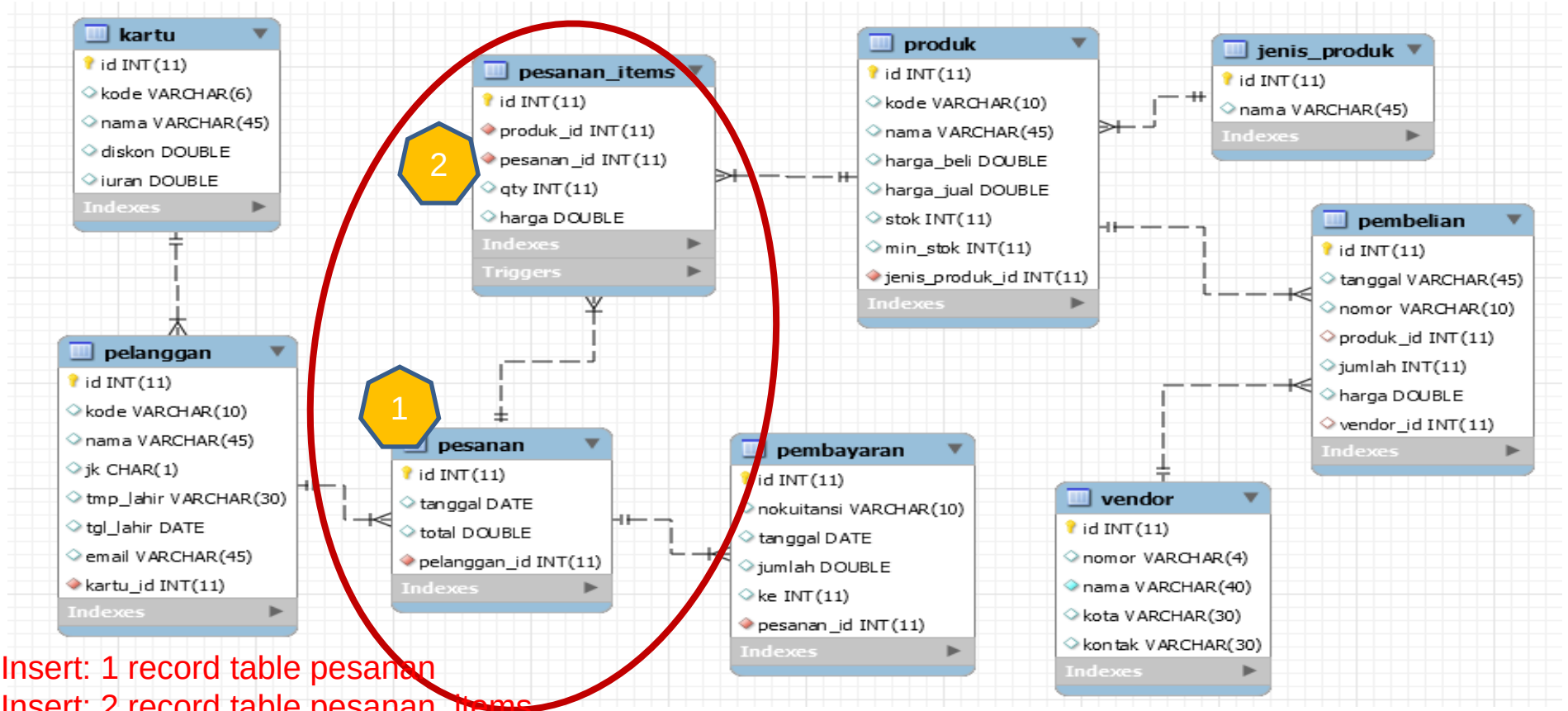
- **Mulai di support MySQL versi 5.0.3, Hanya support untuk Engine innoDB**
- **Sintaks :**

```
mysql> start transaction ;
      BLOK Perintah SQL
mysql> commit/rollback ;
```

Contoh Transaction

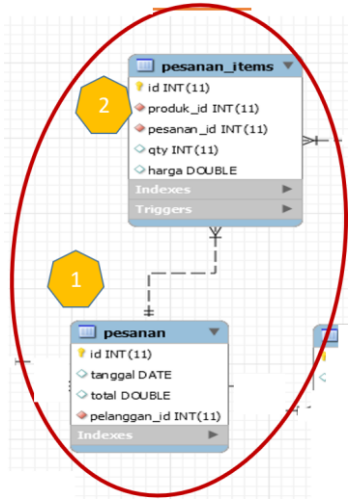
```
mysql> select * from jenis_produk;// cek sebelum transaksi
mysql> start transaction ;// mulai transaksi
mysql> insert into jenis_produk values (default,'smartphone');
mysql> insert into jenis_produk values (default,'ATK');
mysql> UPDATE produk set nama='komputer & jaringan' where id=5;
mysql> select * from jenis_produk;// cek sesudah insert & update
mysql> rollback; // batalkan transaksi
mysql> select * from jenis_produk;// cek apakah data kembali semula
```

Contoh :: Transaksi database pesanan – pesanan_items



Insert: 1 record table pesanan
Insert: 2 record table pesanan_items

Query insert pesanan – pesanan_items




```
mysql> insert into pesanan (tanggal,pelanggan_id) values (current_date,1);
mysql> select * from pesanan where tanggal=current_date;
+----+------------+-------+--------------+
| id | tanggal    | total | pelanggan_id |
+----+------------+-------+--------------+
| 11 | 2021-03-24 |  NULL |            1 |
+----+------------+-------+--------------+
mysql> insert into pesanan_items(produk_id,pesanan_id,qty,harga) values (4,11,1,600000);
mysql> insert into pesanan_items(produk_id,pesanan_id,qty,harga) values (9,11,1,12000000);
```

- Seharusnya jika salah satu query terjadi error (gagal insert /update) maka seharusnya semua proses query dibatalkan
- Rangkaian proses query ini dikenal dengan istilah TRANSACTION, Jika salah satu proses query gagal maka data dikembalikan seperti semula proses ini dinamakan ROLLBACK
- Jika semua query berjalan sukses maka dilakukan COMMIT

Query Insert dengan Transaction

```
mysql> start transaction;
mysql> insert into pesanan (tanggal,pelanggan_id) values
(current_date,1);
mysql> select * from pesanan where tanggal=current_date;
+----+------------+-------+--------------+
| id | tanggal    | total | pelanggan_id |
+----+------------+-------+--------------+
| 11 | 2021-03-24 |  NULL |            1 |
+----+------------+-------+--------------+
mysql> insert into pesanan_items(produk_id,pesanan_id,qty,harga)
values (4,11,1,600000);
mysql> insert into pesanan_items(produk_id,pesanan_id,qty,harga)
values (9,11,1,12000000);
mysql> commit ;
        atau menggagalkan transaksi
mysql> rollback;
```

Transaksi pada program PHP

```
<?php
try {
$dbh = new PDO('mysql:host=localhost;dbname=latihan','oca','ok');
  // memulai transaksi
  $dbh->beginTransaction();

 // query pertama
  $dbh->exec(" insert into pembelian values
('','B001',2,20,900,1300,now())");

 // query kedua
  $dbh->exec(" update produk set jumlah=jumlah + 20,harga=1300 where
idproduk=2");
  // commit transaksi
  $dbh->commit();

} catch (Exception $e) {
  // batalkan transaksi
  $dbh->rollBack();
  echo "Failed: " . $e->getMessage();
}
?>
```

TERIMA KASIH
ATAS SEGALA PERHATIAN
SEMOGA BERMANFAAT...