King Mongkut's University of Technology Thonburi

Project Report Database Design

On

E-commerce System

Submitted by

Banlearit Siriboon No. 64130500105 (50%)

Bowonwit Anothaisintavee No. 64130500105 (50%)

Present

Ajarn. Kittipong Warasup

Ajarn. Sunisa Sathapornvajana

Course INT205 Database Management System Semester 2022

1 November 2022

Table of Contents

Business Requirements _____ *1*

Logical Database Design _____ *2*

SQL Statements _____ *5*

Data Dictionary _____ *8*

DDL Script _____ *13*

Insert Statement _____ *22*

Business Requirements

ระบบจัดเก็บข้อมูลสำหรับเว็บไซต์ e-commerce ที่ต้องการขายผลิตภัณฑ์ประเภท IOT ที่เกี่ยวข้องกับ Smart home

ผู้ใช้จะต้องมีบัญชีผู้ใช้ โดยจะต้องกรอกข้อมูล อีเมล รหัสผ่าน ชื่อ-นามสกุล และเบอร์โทรศัพท์เพื่อสมัครบัญชี ผู้ใช้สามารถเพิ่มข้อมูลที่อยู่ได้ในภายหลัง

ผู้ใช้แต่ละคนจะสามารถสั่งซื้อผลิตภัณฑ์ได้หลายรายการ โดยการเลือกผลิตภัณฑ์เพิ่มเข้าไปในตะกร้าสินค้า ภายในตะกร้าสินค้านั้นผู้ใช้ สามารถเลือกที่จะชำระเงิน เพิ่ม ลด หรือนำผลิตภัณฑ์ออกจากตะกร้าสินค้าได้

หน้าผลิตภัณฑ์จะมีรายละเอียด ประกอบด้วยชื่อสินค้า ราคา คำอธิบายสินค้า ประเภทผลิตภัณฑ์ ป้ายกำกับ ปริมาณสินค้าในคลัง และรายละเอียดคำอธิบาย ซึ่งสินค้าจะมีการเก็บข้อมูลราคาสินค้าย้อนหลังเพื่อใช้ในการวิเคราะห์ข้อมูล รวมถึงราคาสินค้าชั่วคราวเช่น ราคาในช่วงลดราคา

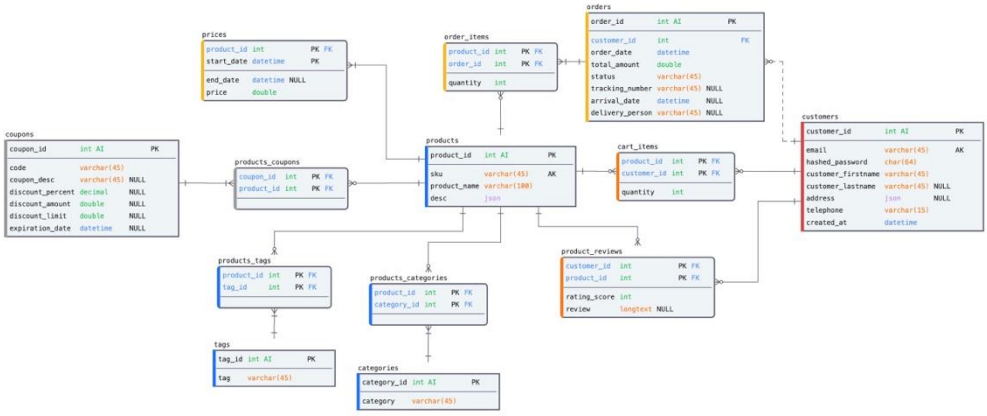
ผลิตภัณฑ์จะแบ่งตามประเภท และแต่ละผลิตภัณฑ์จะมีป้ายกำกับได้มากกว่าหนึ่งอย่าง โดยป้ายกำกับจะเจาะจงชนิดของสินค้าได้มากขึ้น ผลิตภัณฑ์แต่ละชิ้นจะมีคูปองส่วนลดเป็นช่วงเวลาหนึ่ง และผู้ใช้ที่มีคูปองส่วนลดจะสามารถใส่คูปองส่วนลดเพื่อลดราคาผลิตภัณฑ์ที่จุดชำระเงินได้

เมื่อขั้นตอนการชำระเงินเสร็จสิ้น จะได้หมายเลขติดตามผลิตภัณฑ์ และจะมีการจัดส่งผลิตภัณฑ์ตามที่อยู่ของผู้ใช้ โดยผู้จัดส่งจะเข้ารับผลิตภัณฑ์จากผู้ขาย และผู้จัดส่งจะนำผลิตภัณฑ์จัดส่งให้ผู้ซื้อตามลำดับ

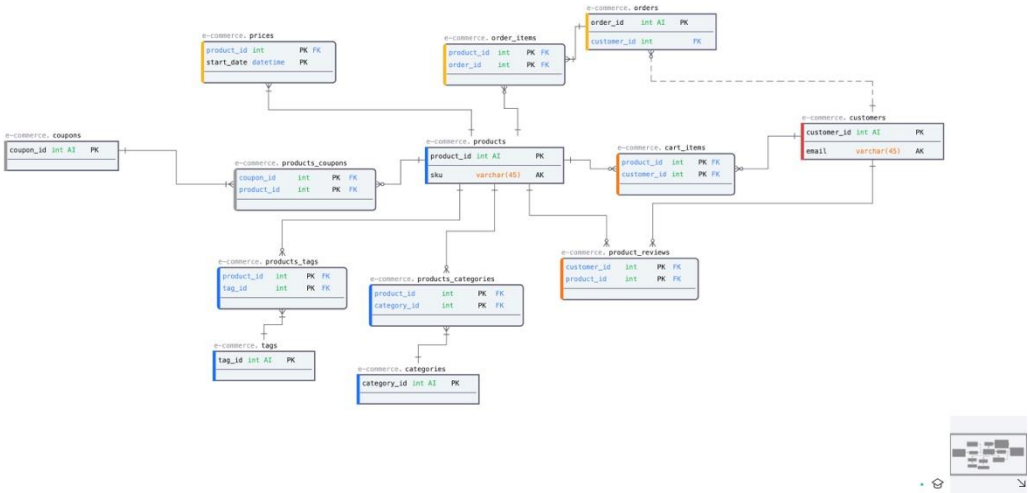
ผู้ซื้อสามารถให้คะแนนและเขียนบทวิจารณ์ให้กับผลิตภัณฑ์ที่ได้ทำการสั่งซื้อ โดยผู้ใช้คนอื่นสามารถเห็นคะแนนและบทวิจารณ์ของผู้ใช้คนอื่นได้

Logical Database Design

Entity Relational View



Keys View



คำอธิบายเพิ่มเติม

ตาราง products

- Attribute product_id จะมี Auto Increment Constrain และมี attribute sku ที่เป็นรหัสจำแนกสินค้าที่มี unique constrain ซึ่ง sku นั้นอาจมีการเปลี่ยนแปลงได้ และต้องมีการเก็บข้อมูลเป็น varchar เนื่องจากรหัสสินค้านั้นอาจประกอบด้วยตัวอักษรและ/หรือตัวเลข การใช้ product_id ซึ่งมีการเก็บข้อมูลแบบ int เป็น primary key จะทำให้ฐานข้อมูลมีประสิทธิภาพที่ดีกว่าการใช้ sku

- Attribute desc เก็บข้อมูลเป็น JSON เนื่องจากแต่ละผลิตภัณฑ์จะมี description หลายส่วนเช่น “overview”, “features”, “what’s Included” หรือ additional information ซึ่งทำให้ข้อมูลรายละเอียดย่อยของผลิตภัณฑ์นั้นจะแยกออกเป็นลำดับขั้นที่ลึกขึ้นตามแต่ละผลิตภัณฑ์

ตาราง prices

- ราคาสินค้าจะมีการเก็บข้อมูลแยกเนื่องจากสินค้านั้นอาจมีการเปลี่ยนแปลงราคาได้ตามกาลเวลาเช่น การปรับราคาตามช่วงเวลา หรือการปรับราคาตามกลยุทธ์การส่งเสริมการขายของบริษัท (promotion)

- Attribute start_date จะเป็นวันที่ ที่มีการบันทึกราคาเริ่มต้น และราคาจะมีผลใช้งานโดยทันที

- Attribute end_date จะเป็น optional ใช้กับราคาชั่วคราวเช่น ราคาจากการส่งเสริมการขาย เมื่อผ่านช่วงเวลาของ end_date ราคาที่แสดงก็จะเป็นราคาล่าสุดที่ไม่ได้มีการจำกัดอายุ end_date ไว้

ตาราง customers

- Attribute address มีการเก็บข้อมูลเป็น JSON เพราะ address อาจประกอบไปด้วย street, city, postal code, country code, country, และ text ซึ่งเป็นข้อมูลที่จำเป็นต่อการจัดส่งสินค้าให้กับลูกค้า

ตาราง coupons

- คูปองแต่ละคูปองนั้นจะสามารถใช้กับผลิตภัณฑ์ได้ตามแต่ละที่คูปองนั้นได้ระบุไว้จึงจำเป็นต้องแยกเก็บข้อมูลของคูปอง

- Attribute discount_percent และ discount_amout นั้นจะใช้สำหรับคูปองที่มีการลดแบบเป็นเปอร์เซ็นต์หรือลดเป็นราคา เช่น ลด 15%, 20%, ลด 300 บาท หรือ ลด 1,000 บาทเป็นต้น โดยที่ discount_percent อาจใช้ร่วมกับ attribute discount_limit เพื่อจำกัดการลดราคาของคูปอง

ตาราง cart_items

- ใช้สำหรับเก็บข้อมูลตะกร้าสินค้าของลูกค้าเพื่อให้สามารถเข้าถึงข้อมูลตะกร้าสินค้าข้ามแพลตฟอร์มที่ลงชื่อเข้าใช้งานได้ เมื่อมีการยืนยันคำสั่งซื้อ ก็จะมีการเคลียร์ตะกร้าสินค้าแล้วนำสินค้าที่สั่งซื้อไปเก็บในตาราง order_items ต่อไป

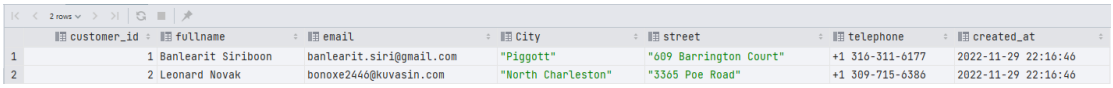
SQL Statements

Description: หา customer_id, fullname, email, city, street, telephone, created_ad ของ User ที่มีการเขียนบทวิจารณ์

SQL Statement:

```
SELECT c.customer_id, CONCAT(c.customer_firstname, ' ', c.customer_lastname) AS
fullname, c.email, c.address->'$.address.City' AS City, c.address->'$.address.street'
AS street, c.telephone, c.created_at
FROM customers c JOIN product_reviews pr ON c.customer_id = pr.customer_id
WHERE pr.review IS NOT NULL;
```

Query Result:



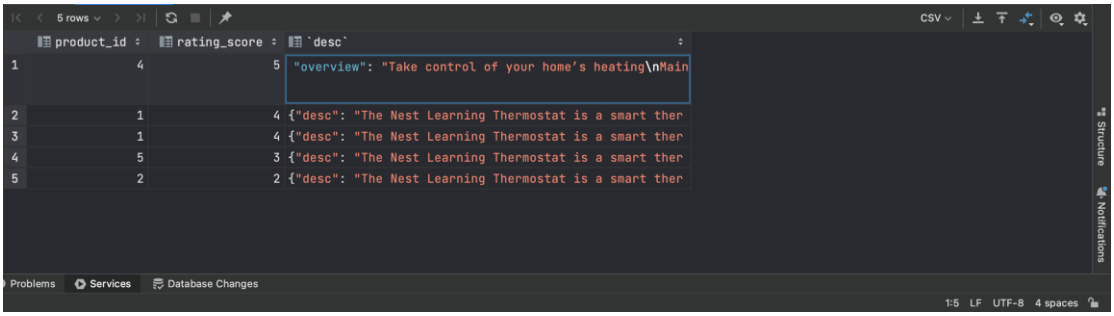
	customer_id	fullname	email	City	street	telephone	created_at
1	1	Banlearit Siriboon	banlearit.siri@gmail.com	"Piggott"	"609 Barrington Court"	+1 316-311-6177	2022-11-29 22:16:46
2	2	Leonard Novak	bonoxe2446@kuvasin.com	"North Charleston"	"3365 Poe Road"	+1 309-715-6386	2022-11-29 22:16:46

Description: เพิ่ม object JSON เข้าไปใน attribute desc ของตาราง products ที่มี rating score สูงสุด

SQL Statement:

```
UPDATE products SET `desc` = JSON_INSERT(`desc`,'$.overview','Take control of your
home’s heating
Maintain a comfortable climate in your home and conserve energy with the Google Nest
Learning Thermostat 3rd Gen Stainless Steel.
It becomes accustomed to your habits and schedule over time, making adjustments
automatically at certain times of the day.
You can also see data received from your individual temperature sensors and raise or lower
the temperature as needed from anywhere
using your mobile device.')
WHERE (SELECT MAX(rating_score)
FROM product_reviews);
SELECT p.product_id, pr.rating_score, p.desc
FROM products p JOIN product_reviews pr USING(product_id)
ORDER BY pr.rating_score DESC;
```

Query Result:



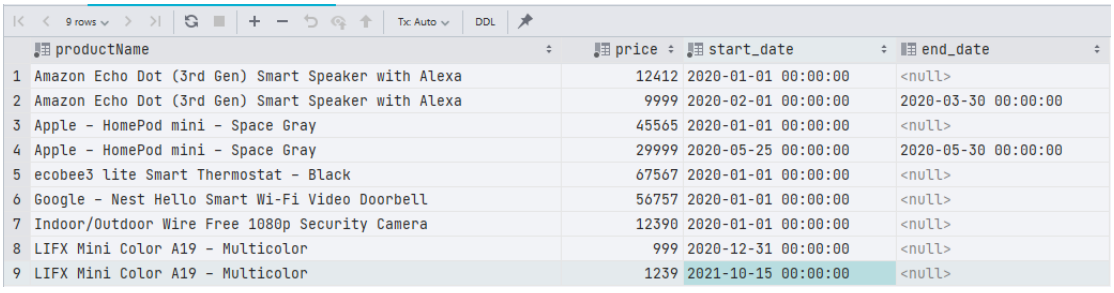
	product_id	rating_score	`desc`
1	4	5	"overview": "Take control of your home's heating\nMain
2	1	4	{"desc": "The Nest Learning Thermostat is a smart ther
3	1	4	{"desc": "The Nest Learning Thermostat is a smart ther
4	5	3	{"desc": "The Nest Learning Thermostat is a smart ther
5	2	2	{"desc": "The Nest Learning Thermostat is a smart ther

Description: สร้าง view ที่ชื่อว่า products_prices_view มีข้อมูล product_name, price, start_date และend_date โดยเรียงลำดับตามชื่อผลิตภัณฑ์และวันที่ที่มีการลงราคาเริ่มต้น

SQL Statement:

```
CREATE OR REPLACE VIEW products_prices_view
(productName,price,start_date,end_date) AS
SELECT product_name,price,start_date,end_date
FROM products JOIN prices p USING(product_id)
ORDER BY product_name, start_date;
SELECT * FROM products_prices_view;
```

Query Result:



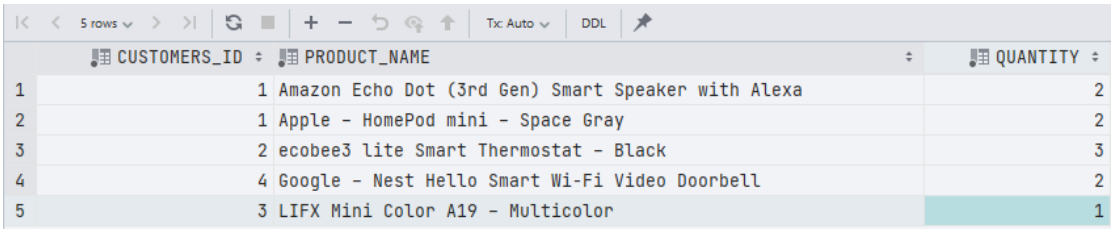
	productName	price	start_date	end_date
1	Amazon Echo Dot (3rd Gen) Smart Speaker with Alexa	12412	2020-01-01 00:00:00	<null>
2	Amazon Echo Dot (3rd Gen) Smart Speaker with Alexa	9999	2020-02-01 00:00:00	2020-03-30 00:00:00
3	Apple - HomePod mini - Space Gray	45565	2020-01-01 00:00:00	<null>
4	Apple - HomePod mini - Space Gray	29999	2020-05-25 00:00:00	2020-05-30 00:00:00
5	ecobee3 lite Smart Thermostat - Black	67567	2020-01-01 00:00:00	<null>
6	Google - Nest Hello Smart Wi-Fi Video Doorbell	56757	2020-01-01 00:00:00	<null>
7	Indoor/Outdoor Wire Free 1080p Security Camera	12390	2020-01-01 00:00:00	<null>
8	LIFX Mini Color A19 - Multicolor	999	2020-12-31 00:00:00	<null>
9	LIFX Mini Color A19 - Multicolor	1239	2021-10-15 00:00:00	<null>

Description: สร้าง view ที่ชื่อว่า delivered_order_items_view โดยแสดงข้อมูล customer_id, product_name, quantity ที่ได้รับ order เป็นที่เรียบร้อยแล้ว

SQL Statement:

```
CREATE OR REPLACE VIEW delivered_order_items_view (customer_id, product_name, quantity)
AS SELECT customer_id, product_name, quantity
FROM customers JOIN (SELECT customer_id, product_name, quantity, arrival_date
                     FROM products JOIN order_items USING(product_id) JOIN orders
USING(order_id)) AS product_quantities_names USING(customer_id)
WHERE DATE(arrival_date) < DATE(NOW());
SELECT * FROM delivered_order_items_view;
SELECT DATE(arrival_date), DATE(NOW())
FROM orders;
```

Query Result:



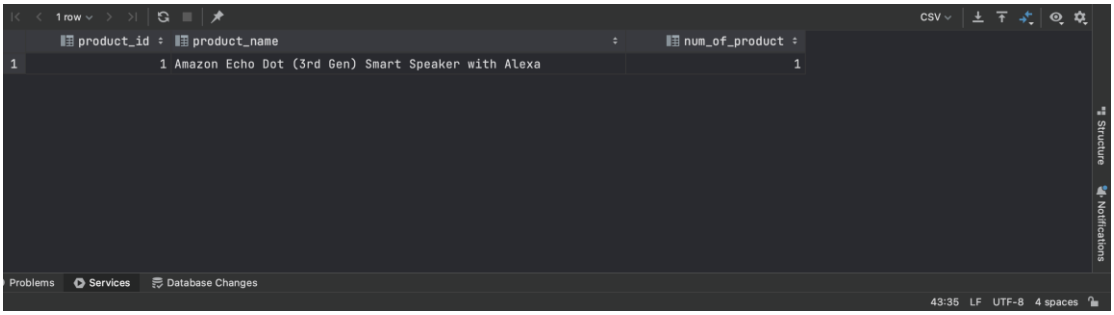
	CUSTOMERS_ID	PRODUCT_NAME	QUANTITY
1	1	Amazon Echo Dot (3rd Gen) Smart Speaker with Alexa	2
2	1	Apple - HomePod mini - Space Gray	2
3	2	ecobee3 lite Smart Thermostat - Black	3
4	4	Google - Nest Hello Smart Wi-Fi Video Doorbell	2
5	3	LIFX Mini Color A19 - Multicolor	1

Description: แสดงจำนวนของ product ที่มีข้อมูล overview ใน JSON ของ desc

SQL Statement:

```
SELECT product_id, product_name, COUNT(product_id) AS num_of_product
FROM products
WHERE JSON_EXTRACT(`desc`, '$.overview') IS NOT NULL
GROUP BY product_id, product_name;
```

Query Result:



	product_id	product_name	num_of_product
1	1	Amazon Echo Dot (3rd Gen) Smart Speaker with Alexa	1

Data Dictionary

Table: products

Description: ข้อมูลของผลิตภัณฑ์

No.	Column Name	Description	Data Type	Constraint	Default Value
1	product_id	หมายเลขผลิตภัณฑ์	INT	PK, AI	Next Sequential Number
2	sku	รหัสสินค้าที่มีการจำแนกประเภทสินค้าได้ตามความแตกต่างของสินค้า	VARCHAR(45)	NN, UQ	
3	product_name	ชื่อผลิตภัณฑ์	VARCHAR(100)	NN	
4	desc	คำอธิบายผลิตภัณฑ์	JSON	NN	

Table: categories

Description: ประเภทผลิตภัณฑ์

No.	Column Name	Description	Data Type	Constraint	Default Value
1	categories_id	หมายเลขประเภทผลิตภัณฑ์	INT	PK, NN, AI	Next Sequential Number
2	category	ประเภทของผลิตภัณฑ์	VARCHAR(45)	NN	

Table: tags

Description: ป้ายกำกับ

No.	Column Name	Description	Data Type	Constraint	Default Value
1	tag_id	หมายเลขป้ายกำกับ	INT	PK, NN, AI	Next Sequential Number
2	tag	ป้ายกำกับผลิตภัณฑ์	VARCHAR(45)	NN	

Table: products_tags

Description: ป้ายกับผลิตภัณฑ์

No.	Column Name	Description	Data Type	Constraint	Referenced Table
1	product_id	หมายเลขผลิตภัณฑ์	INT	PK, NN, FK	products
2	tag_id	หมายเลขป้ายกำกับ	INT	PK, NN, FK	tags

Table: products_categories

Description: ประเภทของผลิตภัณฑ์

No.	Column Name	Description	Data Type	Constraint	Referenced Table
1	product_id	หมายเลขผลิตภัณฑ์	INT	PK, NN, FK	products
2	category_id	หมายเลขประเภทผลิตภัณฑ์	INT	PK, NN, FK	categories

Table: coupons

Description: คูปองส่วนลด

No.	Column Name	Description	Data Type	Constraint	Default Value
1	coupon_id	หมายเลขคูปอง	INT	PK, NN, AI	Next Sequential Number
2	code	รหัสคูปอง	VARCHAR(45)	NN	
3	coupon_desc	ละเอียดคูปอง	VARCHAR(45)		
4	discount_percent	ส่วนลดเป็นเปอร์เซ็นต์	DECIMAL		
5	discount_amount	ส่วนลดเป็นจำนวน	DOUBLE		
6	discount_limit	ส่วนลดสูงสุดจำกัด	DOUBLE		
7	expiration_date	อายุของคูปอง	DATETIME		

Table: products_coupons

Description: คูปองส่วนลดผลิตภัณฑ์

No.	Column Name	Description	Data Type	Constraint	Referenced Table
1	coupon_id	หมายเลขคูปอง	INT	PK, NN, FK	coupons
2	product_id	หมายเลขผลิตภัณฑ์	INT	PK, NN, FK	products

Table: customers

Description: ข้อมูลผู้ใช้

No.	Column Name	Description	Data Type	Constraint	Default Value
1	customer_id	หมายเลขผู้ใช้	INT	PK, NN, AI	Next Sequential Number
2	email	อีเมลผู้ใช้	VARCHAR(45)	NN, UQ	
3	hashed_password	รหัสผ่านผู้ใช้	CHAR(60)	NN	
4	customer_firstname	ชื่อผู้ใช้	VARCHAR(45)	NN	
5	customer_lastname	นามสกุลผู้ใช้	VARCHAR(45)		
6	address	ข้อมูลที่อยู่ผู้ใช้	JSON		
7	telephone	เบอร์โทรศัพท์ผู้ใช้	VARCHAR(15)	NN	
8	created_at	เวลาที่ผู้ใช้สร้างบัญชี	DATETIME	NN	CURRENT_TIMESTAMP

Table: order

Description: ข้อมูลการสั่งซื้อ

No.	Column Name	Description	Data Type	Constraint	Referenced Table	Default Value
1	order_id	หมายเลขคำสั่งซื้อ	INT	PK, NN, AI		Next Sequential Number
2	customer_id	หมายเลขผู้ใช้	INT	NN, FK	customers	
3	order_date	เวลาสั่งซื้อ	DATETIME	NN		CURRENT_TIMESTAMP
4	total_amount	จำนวนการสั่งซื้อทั้งหมด	DOUBLE	NN		
5	status	สถานะสั่งซื้อ	VARCHAR(45)	NN		
6	tracking_number	หมายเลขติดตามวัสดุ	VARCHAR(45)			
7	arrival_date	วันที่วัสดุถึง	DATETIME			
8	delivery_person	คนที่จัดส่งวัสดุ	VARCHAR(45)			

Table: order_items

Description: รายการสั่งซื้อ

No.	Column Name	Description	Data Type	Constraint	Referenced Table
1	product_id	หมายเลขผลิตภัณฑ์	INT	PK, NN, FK	products
2	order_id	หมายเลขคำสั่งซื้อ	INT	PK, NN, FK	orders
3	quantity	จำนวนที่สั่งซื้อ	INT	NN	

Table: cart_items

Description: รายการสั่งซื้อในตะกร้า

No.	Column Name	Description	Data Type	Constraint	Referenced Table
1	product_id	หมายเลขผลิตภัณฑ์	INT	PK, NN, FK	products
2	customer_id	หมายเลขคำสั่งซื้อ	INT	PK, NN, FK	orders
3	quantity	จำนวนที่สั่งซื้อ	INT	NN	

Table: product_reviews

Description: คำวิจารณ์ผลิตภัณฑ์

No.	Column Name	Description	Data Type	Constraint	Referenced Table
1	customer_id	หมายเลขคำสั่งซื้อ	INT	PK,NN,FK	
2	product_id	หมายเลขผลิตภัณฑ์	INT	PK,NN,FK	
3	rating_score	การถูกใจของผลิตภัณฑ์	INT	NN	
4	review	คำวิจารณ์	LONGTEXT		

Table: prices

Description: ราคาสินค้า

No.	Column Name	Description	Data Type	Constraint	Referenced Table
1	product_id	หมายเลขผลิตภัณฑ์	INT	PK,NN,FK	products
2	start_date	วันที่กำหนดราคาผลิตภัณฑ์	DATETIME	PK,NN	
3	end_date	วันที่ล่าสุดของราคาผลิตภัณฑ์	DATETIME		
4	price	ราคาผลิตภัณฑ์	DOUBLE	NN	

DDL Script

```
SET @OLD_UNIQUE_CHECKS=@@UNIQUE_CHECKS, UNIQUE_CHECKS=0;
SET @OLD_FOREIGN_KEY_CHECKS=@@FOREIGN_KEY_CHECKS,
FOREIGN_KEY_CHECKS=0;
SET @OLD_SQL_MODE=@@SQL_MODE,
SQL_MODE='ONLY_FULL_GROUP_BY,STRICT_TRANS_TABLES,NO_ZERO_IN_DAT
E,NO_ZERO_DATE,ERROR_FOR_DIVISION_BY_ZERO,NO_ENGINE_SUBSTITUTION';


-- -----------------------------------------------
-- Schema e-commerce
-- -----------------------------------------------
DROP SCHEMA IF EXISTS `e-commerce` ;


-- -----------------------------------------------
-- Schema e-commerce
-- -----------------------------------------------
CREATE SCHEMA IF NOT EXISTS `e-commerce` ;
USE `e-commerce` ;


-- -----------------------------------------------
-- Table `e-commerce`.`products`
-- -----------------------------------------------
DROP TABLE IF EXISTS `e-commerce`.`products` ;

CREATE TABLE IF NOT EXISTS `e-commerce`.`products` (
  `product_id` INT NOT NULL AUTO_INCREMENT,
  `sku` VARCHAR(45) NOT NULL,
  `product_name` VARCHAR(100) NOT NULL,
  `desc` JSON NOT NULL,
  PRIMARY KEY (`product_id`),
```

```
  UNIQUE INDEX `UX__PRODUCTS__SKU` (`sku` ASC) VISIBLE)
ENGINE = InnoDB;


-- -----------------------------------------------
-- Table `e-commerce`.`prices`
-- -----------------------------------------------
DROP TABLE IF EXISTS `e-commerce`.`prices` ;

CREATE TABLE IF NOT EXISTS `e-commerce`.`prices` (
  `product_id` INT NOT NULL,
  `start_date` DATETIME NOT NULL,
  `end_date` DATETIME NULL,
  `price` DOUBLE NOT NULL,
  PRIMARY KEY (`product_id`, `start_date`),
  INDEX `FK__PRICES__PRODUCT_ID` (`product_id` ASC) VISIBLE,
  CONSTRAINT `FK__PRICES__PRODUCTS`
    FOREIGN KEY (`product_id`)
    REFERENCES `e-commerce`.`products` (`product_id`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION)
ENGINE = InnoDB;


-- -----------------------------------------------
-- Table `e-commerce`.`tags`
-- -----------------------------------------------
DROP TABLE IF EXISTS `e-commerce`.`tags` ;

CREATE TABLE IF NOT EXISTS `e-commerce`.`tags` (
  `tag_id` INT NOT NULL AUTO_INCREMENT,
  `tag` VARCHAR(45) NOT NULL,
  PRIMARY KEY (`tag_id`))
```

```
ENGINE = InnoDB;


-- -----------------------------------------------
-- Table `e-commerce`.`categories`
-- -----------------------------------------------
DROP TABLE IF EXISTS `e-commerce`.`categories` ;

CREATE TABLE IF NOT EXISTS `e-commerce`.`categories` (
  `category_id` INT NOT NULL AUTO_INCREMENT,
  `category` VARCHAR(45) NOT NULL,
  PRIMARY KEY (`category_id`))
ENGINE = InnoDB;


-- -----------------------------------------------
-- Table `e-commerce`.`products_tags`
-- -----------------------------------------------
DROP TABLE IF EXISTS `e-commerce`.`products_tags` ;

CREATE TABLE IF NOT EXISTS `e-commerce`.`products_tags` (
  `product_id` INT NOT NULL,
  `tag_id` INT NOT NULL,
  PRIMARY KEY (`product_id`, `tag_id`),
  INDEX `FK__PRODUCTS_TAGS__PRODUCT_ID` (`product_id` ASC) VISIBLE,
  INDEX `FK__PRODUCTS_TAGS__TAG_ID` (`tag_id` ASC) VISIBLE,
  CONSTRAINT `FK__PRODUCTS_TAGS__TAGS`
    FOREIGN KEY (`tag_id`)
    REFERENCES `e-commerce`.`tags` (`tag_id`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION,
  CONSTRAINT `FK__PRODUCTS_TAGS__PRODUCTS`
    FOREIGN KEY (`product_id`)
```

```
    REFERENCES `e-commerce`.`products` (`product_id`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION)
ENGINE = InnoDB;


-- -----------------------------------------------
-- Table `e-commerce`.`products_categories`
-- -----------------------------------------------
DROP TABLE IF EXISTS `e-commerce`.`products_categories` ;


CREATE TABLE IF NOT EXISTS `e-commerce`.`products_categories` (
  `product_id` INT NOT NULL,
  `category_id` INT NOT NULL,
  PRIMARY KEY (`product_id`, `category_id`),
  INDEX `FK__PRODUCTS_CATEGORIES__PRODUCT_ID` (`product_id` ASC)
VISIBLE,
  INDEX `FK__PRODUCTS_CATEGORIES__CATEGORY_ID` (`category_id` ASC)
VISIBLE,
  CONSTRAINT `FK__PRODUCTS_CATEGORIES__CATEGORIES`
    FOREIGN KEY (`category_id`)
    REFERENCES `e-commerce`.`categories` (`category_id`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION,
  CONSTRAINT `FK__PRODUCTS_CATEGORIES__PRODUCTS`
    FOREIGN KEY (`product_id`)
    REFERENCES `e-commerce`.`products` (`product_id`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION)
ENGINE = InnoDB;


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

```
-- Table `e-commerce`.`customers`
-- -----------------------------------------------
DROP TABLE IF EXISTS `e-commerce`.`customers` ;

CREATE TABLE IF NOT EXISTS `e-commerce`.`customers` (
  `customer_id` INT NOT NULL AUTO_INCREMENT,
  `email` VARCHAR(45) NOT NULL,
  `hashed_password` CHAR(64) NOT NULL,
  `customer_firstname` VARCHAR(45) NOT NULL,
  `customer_lastname` VARCHAR(45) NULL,
  `address` JSON NULL,
  `telephone` VARCHAR(15) NOT NULL,
  `created_at` DATETIME NOT NULL DEFAULT CURRENT_TIMESTAMP,
  PRIMARY KEY (`customer_id`),
  UNIQUE INDEX `UX__CUSTOMERS__EMAIL` (`email` ASC) VISIBLE)
ENGINE = InnoDB;


-- -----------------------------------------------
-- Table `e-commerce`.`cart_items`
-- -----------------------------------------------
DROP TABLE IF EXISTS `e-commerce`.`cart_items` ;

CREATE TABLE IF NOT EXISTS `e-commerce`.`cart_items` (
  `product_id` INT NOT NULL,
  `customer_id` INT NOT NULL,
  `quantity` INT NOT NULL,
  PRIMARY KEY (`product_id`, `customer_id`),
  INDEX `FK__CART_ITEMS__CUSTOMER_ID` (`customer_id` ASC) VISIBLE,
  INDEX `FK__CART_ITEMS__PRODUCT_ID` (`product_id` ASC) VISIBLE,
  CONSTRAINT `FK__CART_ITEMS__PRODUCTS`
    FOREIGN KEY (`product_id`)
```

```
    REFERENCES `e-commerce`.`products` (`product_id`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION,
  CONSTRAINT `FK__CART_ITEMS__CUSTOMERS`
    FOREIGN KEY (`customer_id`)
    REFERENCES `e-commerce`.`customers` (`customer_id`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION)
ENGINE = InnoDB;


-- -----------------------------------------------------
-- Table `e-commerce`.`orders`
-- -----------------------------------------------------
DROP TABLE IF EXISTS `e-commerce`.`orders` ;


CREATE TABLE IF NOT EXISTS `e-commerce`.`orders` (
  `order_id` INT NOT NULL AUTO_INCREMENT,
  `customer_id` INT NOT NULL,
  `order_date` DATETIME NOT NULL DEFAULT CURRENT_TIMESTAMP,
  `total_amount` DOUBLE NOT NULL,
  `status` VARCHAR(45) NOT NULL,
  `tracking_number` VARCHAR(45) NULL,
  `arrival_date` DATETIME NULL,
  `delivery_person` VARCHAR(45) NULL,
  PRIMARY KEY (`order_id`),
  CONSTRAINT `FK__ORDERS__CUSTOMERS`
    FOREIGN KEY (`customer_id`)
    REFERENCES `e-commerce`.`customers` (`customer_id`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION)
ENGINE = InnoDB;
```

```
-- -----------------------------------------------
-- Table `e-commerce`.`order_items`
-- -----------------------------------------------
DROP TABLE IF EXISTS `e-commerce`.`order_items` ;

CREATE TABLE IF NOT EXISTS `e-commerce`.`order_items` (
  `product_id` INT NOT NULL,
  `order_id` INT NOT NULL,
  `quantity` INT NOT NULL,
  PRIMARY KEY (`product_id`, `order_id`),
  INDEX `FK__ORDER_ITEMS__ORDER_ID` (`order_id` ASC) VISIBLE,
  INDEX `FK__ORDER_ITEMS__PRODUCT_ID` (`product_id` ASC) VISIBLE,
  CONSTRAINT `FK__ORDER_ITEMS__PRODUCTS`
    FOREIGN KEY (`product_id`)
    REFERENCES `e-commerce`.`products` (`product_id`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION,
  CONSTRAINT `FK__ORDER_ITEMS__ORDERS`
    FOREIGN KEY (`order_id`)
    REFERENCES `e-commerce`.`orders` (`order_id`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION)
ENGINE = InnoDB;


-- -----------------------------------------------
-- Table `e-commerce`.`product_reviews`
-- -----------------------------------------------
DROP TABLE IF EXISTS `e-commerce`.`product_reviews` ;

CREATE TABLE IF NOT EXISTS `e-commerce`.`product_reviews` (
```

```
  `customer_id` INT NOT NULL,
  `product_id` INT NOT NULL,
  `rating_score` INT NOT NULL,
  `review` LONGTEXT NULL,
  PRIMARY KEY (`customer_id`, `product_id`),
  INDEX `FK__PRODUCT_REVIEWS__PRODUCT_ID` (`product_id` ASC) VISIBLE,
  INDEX `FK__PRODUCT_REVIEWS__CUSTOMER_ID` (`customer_id` ASC) VISIBLE,
  CONSTRAINT `FK__PRODUCT_REVIEWS__CUSTOMERS`
    FOREIGN KEY (`customer_id`)
    REFERENCES `e-commerce`.`customers` (`customer_id`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION,
  CONSTRAINT `FK__PRODUCT_REVIEWS__PRODUCTS`
    FOREIGN KEY (`product_id`)
    REFERENCES `e-commerce`.`products` (`product_id`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION)
ENGINE = InnoDB;


-- -----------------------------------------------
-- Table `e-commerce`.`coupons`
-- -----------------------------------------------
DROP TABLE IF EXISTS `e-commerce`.`coupons` ;

CREATE TABLE IF NOT EXISTS `e-commerce`.`coupons` (
  `coupon_id` INT NOT NULL AUTO_INCREMENT,
  `code` VARCHAR(45) NOT NULL,
  `coupon_desc` VARCHAR(45) NULL,
  `discount_percent` DECIMAL NULL,
  `discount_amount` DOUBLE NULL,
  `discount_limit` DOUBLE NULL,
```

```
  `expiration_date` DATETIME NULL,
  PRIMARY KEY (`coupon_id`))
ENGINE = InnoDB;


-- -----------------------------------------------
-- Table `e-commerce`.`products_coupons`
-- -----------------------------------------------
DROP TABLE IF EXISTS `e-commerce`.`products_coupons` ;

CREATE TABLE IF NOT EXISTS `e-commerce`.`products_coupons` (
  `coupon_id` INT NOT NULL,
  `product_id` INT NOT NULL,
  PRIMARY KEY (`coupon_id`, `product_id`),
  INDEX `FK__PRODUCTS_COUPONS__PRODUCT_ID` (`product_id` ASC) VISIBLE,
  INDEX `FK__PRODUCTS_COUPONS__COUPON_ID` (`coupon_id` ASC) VISIBLE,
  CONSTRAINT `FK__PRODUCTS_COUPONS__COUPONS`
    FOREIGN KEY (`coupon_id`)
    REFERENCES `e-commerce`.`coupons` (`coupon_id`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION,
  CONSTRAINT `FK__PRODUCTS_COUPONS__PRODUCTS`
    FOREIGN KEY (`product_id`)
    REFERENCES `e-commerce`.`products` (`product_id`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION)
ENGINE = InnoDB;


SET SQL_MODE=@OLD_SQL_MODE;
SET FOREIGN_KEY_CHECKS=@OLD_FOREIGN_KEY_CHECKS;
SET UNIQUE_CHECKS=@OLD_UNIQUE_CHECKS;
```

Insert Statement

```
-- -----------------------------------------------------
-- Data for table `e-commerce`.`products`
-- -----------------------------------------------------
START TRANSACTION;
USE `e-commerce`;
INSERT INTO `e-commerce`.`products` (`sku`, `product_name`, `desc`) VALUES ('53870', 'Amazon Echo Dot (3rd Gen) Smart Speaker with Alexa', '{\"desc\": \"The Nest Learning Thermostat is a smart thermostat that learns your schedule and programs itself to help save energy. You can control it from anywhere with the Nest app, and it works with Alexa and Google Assistant so you can adjust the temperature with your voice. And it’s beautifully designed, with a big, bright display.\"}');
INSERT INTO `e-commerce`.`products` (`sku`, `product_name`, `desc`) VALUES ('34589', 'Apple – HomePod mini – Space Gray', '{\"desc\": \"The Nest Learning Thermostat is a smart thermostat that learns your schedule and programs itself to help save energy. You can control it from anywhere with the Nest app, and it works with Alexa and Google Assistant so you can adjust the temperature with your voice. And it’s beautifully designed, with a big, bright display.\"}');
INSERT INTO `e-commerce`.`products` (`sku`, `product_name`, `desc`) VALUES ('18340', 'ecobee3 lite Smart Thermostat – Black', '{\"desc\": \"The Nest Learning Thermostat is a smart thermostat that learns your schedule and programs itself to help save energy. You can control it from anywhere with the Nest app, and it works with Alexa and Google Assistant so you can adjust the temperature with your voice. And it’s beautifully designed, with a big, bright display.\"}');
INSERT INTO `e-commerce`.`products` (`sku`, `product_name`, `desc`) VALUES ('12378', 'Google – Nest Hello Smart Wi-Fi Video Doorbell', '{\"desc\": \"The Nest Learning Thermostat is a smart thermostat that learns your schedule and programs itself to help save energy. You can control it from anywhere with the Nest app,
```

```
and it works with Alexa and Google Assistant so you can adjust the temperature with your voice. And it’s beautifully designed, with a big, bright display.\"}');
INSERT INTO `e-commerce`.`products` (`sku`, `product_name`, `desc`) VALUES ('34545', 'Indoor/Outdoor Wire Free 1080p Security Camera', '{\"desc\": \"The Nest Learning Thermostat is a smart thermostat that learns your schedule and programs itself to help save energy. You can control it from anywhere with the Nest app, and it works with Alexa and Google Assistant so you can adjust the temperature with your voice. And it’s beautifully designed, with a big, bright display.\"}');
INSERT INTO `e-commerce`.`products` (`sku`, `product_name`, `desc`) VALUES ('23458', 'LIFX Mini Color A19 – Multicolor', '{\"desc\": \"The Nest Learning Thermostat is a smart thermostat that learns your schedule and programs itself to help save energy. You can control it from anywhere with the Nest app, and it works with Alexa and Google Assistant so you can adjust the temperature with your voice. And it’s beautifully designed, with a big, bright display.\"}');

COMMIT;


-- ----------------------------------------------------
-- Data for table `e-commerce`.`prices`
-- ----------------------------------------------------
START TRANSACTION;
USE `e-commerce`;
INSERT INTO `e-commerce`.`prices` (`product_id`, `start_date`, `end_date`, `price`) VALUES (1, '2020-01-01', NULL, 12412);
INSERT INTO `e-commerce`.`prices` (`product_id`, `start_date`, `end_date`, `price`) VALUES (2, '2020-01-01', NULL, 45565);
INSERT INTO `e-commerce`.`prices` (`product_id`, `start_date`, `end_date`, `price`) VALUES (3, '2020-01-01', NULL, 67567);
INSERT INTO `e-commerce`.`prices` (`product_id`, `start_date`, `end_date`, `price`) VALUES (4, '2020-01-01', NULL, 56757);
```

```
INSERT INTO `e-commerce`.`prices` (`product_id`, `start_date`, `end_date`, `price`)
VALUES (5, '2020-01-01', NULL, 12390);
INSERT INTO `e-commerce`.`prices` (`product_id`, `start_date`, `end_date`, `price`)
VALUES (6, '2021-10-15', NULL, 1239);
INSERT INTO `e-commerce`.`prices` (`product_id`, `start_date`, `end_date`, `price`)
VALUES (1, '2020-02-01', '2020-03-30', 9999);
INSERT INTO `e-commerce`.`prices` (`product_id`, `start_date`, `end_date`, `price`)
VALUES (2, '2020-05-25', '2020-05-30', 29999);
INSERT INTO `e-commerce`.`prices` (`product_id`, `start_date`, `end_date`, `price`)
VALUES (6, '2020-12-31', NULL, 999);

COMMIT;


-- -----------------------------------------------------
-- Data for table `e-commerce`.`tags`
-- -----------------------------------------------------
START TRANSACTION;
USE `e-commerce`;
INSERT INTO `e-commerce`.`tags` (`tag`) VALUES ('Nest');
INSERT INTO `e-commerce`.`tags` (`tag`) VALUES ('Steel');
INSERT INTO `e-commerce`.`tags` (`tag`) VALUES ('Thermostat');

COMMIT;


-- -----------------------------------------------------
-- Data for table `e-commerce`.`categories`
-- -----------------------------------------------------
START TRANSACTION;
USE `e-commerce`;
INSERT INTO `e-commerce`.`categories` (`category`) VALUES ('Accessories');
INSERT INTO `e-commerce`.`categories` (`category`) VALUES ('Assistants');
```

```
INSERT INTO `e-commerce`.`categories` (`category`) VALUES ('Cameras');
INSERT INTO `e-commerce`.`categories` (`category`) VALUES ('Category Grid');
INSERT INTO `e-commerce`.`categories` (`category`) VALUES ('Heating');
INSERT INTO `e-commerce`.`categories` (`category`) VALUES ('Lighting');
INSERT INTO `e-commerce`.`categories` (`category`) VALUES ('Speakers');
INSERT INTO `e-commerce`.`categories` (`category`) VALUES ('Thermostats');
INSERT INTO `e-commerce`.`categories` (`category`) VALUES ('Security');
INSERT INTO `e-commerce`.`categories` (`category`) VALUES ('Doorbells');

COMMIT;


-- -----------------------------------------------------
-- Data for table `e-commerce`.`products_tags`
-- -----------------------------------------------------
START TRANSACTION;
USE `e-commerce`;
INSERT INTO `e-commerce`.`products_tags` (`product_id`, `tag_id`) VALUES (1, 1);
INSERT INTO `e-commerce`.`products_tags` (`product_id`, `tag_id`) VALUES (1, 2);
INSERT INTO `e-commerce`.`products_tags` (`product_id`, `tag_id`) VALUES (1, 3);
INSERT INTO `e-commerce`.`products_tags` (`product_id`, `tag_id`) VALUES (2, 1);
INSERT INTO `e-commerce`.`products_tags` (`product_id`, `tag_id`) VALUES (2, 2);
INSERT INTO `e-commerce`.`products_tags` (`product_id`, `tag_id`) VALUES (2, 3);
INSERT INTO `e-commerce`.`products_tags` (`product_id`, `tag_id`) VALUES (3, 1);
INSERT INTO `e-commerce`.`products_tags` (`product_id`, `tag_id`) VALUES (3, 2);
INSERT INTO `e-commerce`.`products_tags` (`product_id`, `tag_id`) VALUES (3, 3);
INSERT INTO `e-commerce`.`products_tags` (`product_id`, `tag_id`) VALUES (4, 1);
INSERT INTO `e-commerce`.`products_tags` (`product_id`, `tag_id`) VALUES (4, 2);
INSERT INTO `e-commerce`.`products_tags` (`product_id`, `tag_id`) VALUES (4, 3);
INSERT INTO `e-commerce`.`products_tags` (`product_id`, `tag_id`) VALUES (5, 1);
INSERT INTO `e-commerce`.`products_tags` (`product_id`, `tag_id`) VALUES (5, 2);
INSERT INTO `e-commerce`.`products_tags` (`product_id`, `tag_id`) VALUES (5, 3);
```

```
INSERT INTO `e-commerce`.`products_tags` (`product_id`, `tag_id`) VALUES (6, 1);
INSERT INTO `e-commerce`.`products_tags` (`product_id`, `tag_id`) VALUES (6, 2);
INSERT INTO `e-commerce`.`products_tags` (`product_id`, `tag_id`) VALUES (6, 3);

COMMIT;


-- -----------------------------------------------------
-- Data for table `e-commerce`.`products_categories`
-- -----------------------------------------------------
START TRANSACTION;
USE `e-commerce`;
INSERT INTO `e-commerce`.`products_categories` (`product_id`, `category_id`)
VALUES (1, 1);
INSERT INTO `e-commerce`.`products_categories` (`product_id`, `category_id`)
VALUES (1, 2);
INSERT INTO `e-commerce`.`products_categories` (`product_id`, `category_id`)
VALUES (1, 3);
INSERT INTO `e-commerce`.`products_categories` (`product_id`, `category_id`)
VALUES (1, 4);
INSERT INTO `e-commerce`.`products_categories` (`product_id`, `category_id`)
VALUES (1, 5);
INSERT INTO `e-commerce`.`products_categories` (`product_id`, `category_id`)
VALUES (1, 6);
INSERT INTO `e-commerce`.`products_categories` (`product_id`, `category_id`)
VALUES (1, 7);
INSERT INTO `e-commerce`.`products_categories` (`product_id`, `category_id`)
VALUES (1, 8);
INSERT INTO `e-commerce`.`products_categories` (`product_id`, `category_id`)
VALUES (2, 1);
INSERT INTO `e-commerce`.`products_categories` (`product_id`, `category_id`)
VALUES (3, 2);
```

```
INSERT INTO `e-commerce`.`products_categories` (`product_id`, `category_id`)
VALUES (3, 3);
INSERT INTO `e-commerce`.`products_categories` (`product_id`, `category_id`)
VALUES (3, 4);
INSERT INTO `e-commerce`.`products_categories` (`product_id`, `category_id`)
VALUES (3, 5);
INSERT INTO `e-commerce`.`products_categories` (`product_id`, `category_id`)
VALUES (3, 6);
INSERT INTO `e-commerce`.`products_categories` (`product_id`, `category_id`)
VALUES (3, 9);
INSERT INTO `e-commerce`.`products_categories` (`product_id`, `category_id`)
VALUES (3, 7);
INSERT INTO `e-commerce`.`products_categories` (`product_id`, `category_id`)
VALUES (3, 8);
INSERT INTO `e-commerce`.`products_categories` (`product_id`, `category_id`)
VALUES (4, 9);
INSERT INTO `e-commerce`.`products_categories` (`product_id`, `category_id`)
VALUES (5, 10);
INSERT INTO `e-commerce`.`products_categories` (`product_id`, `category_id`)
VALUES (5, 9);
INSERT INTO `e-commerce`.`products_categories` (`product_id`, `category_id`)
VALUES (6, 1);
INSERT INTO `e-commerce`.`products_categories` (`product_id`, `category_id`)
VALUES (6, 6);

COMMIT;


-- -----------------------------------------------------
-- Data for table `e-commerce`.`customers`
-- -----------------------------------------------------
START TRANSACTION;
```

```
USE `e-commerce`;
INSERT INTO `e-commerce`.`customers` (`email`, `hashed_password`,
`customer_firstname`, `customer_lastname`, `address`, `telephone`) VALUES
('banlearit.siri@gmail.com',
'e82146238410edb61a4bebaaaf79b177e98d5bf572f5089fa42cb38ae34d2dfa',
'Banlearit', 'Siriboon', '{\"address\": {\"street\": \"609 Barrington Court\", \"City\":
\"Piggott\"}}', '+1 316-311-6177');
INSERT INTO `e-commerce`.`customers` (`email`, `hashed_password`,
`customer_firstname`, `customer_lastname`, `address`, `telephone`) VALUES
('bonoxe2446@kuvasin.com',
'0d1e5fab139fe6b34fad464f7e7fc59e9d2eff9a761932d66be5f9b9f8c3c442',
'Leonard', 'Novak', '{\"address\": {\"street\": \"3365 Poe Road\", \"City\": \"North
Charleston\"}}', '+1 309-715-6386');
INSERT INTO `e-commerce`.`customers` (`email`, `hashed_password`,
`customer_firstname`, `customer_lastname`, `address`, `telephone`) VALUES
('dasole8187@nubotel.com',
'a0ac8f2ba8dab76a64dd989a421997a6d8d89e00dde7ada20a00eac48e44d508',
'Shaylee', 'Estrada', '{\"address\": { \"street\": \"1863 Hemlock Lane\", \"City\":
\"Harlingen\"}}', '+1 505-644-2303');
INSERT INTO `e-commerce`.`customers` (`email`, `hashed_password`,
`customer_firstname`, `customer_lastname`, `address`, `telephone`) VALUES
('cotob97508@nubotel.com',
'0550566a8aae53578df9db287a6beb840f82dea9fe364de0ee861408c607e48b',
'Yael', 'Hayes', '{\"address\": { \"street\": \"622 Shady Pines Drive\", \"City\":
\"Wytheville\"}}', '+1 224-411-2818');
INSERT INTO `e-commerce`.`customers` (`email`, `hashed_password`,
`customer_firstname`, `customer_lastname`, `address`, `telephone`) VALUES
('deyicof678@nubotel.com',
'fb511352960b627340a8e66f991b8c5f43f9f1de2a1bbe9646fc1ba52b36b31a', 'Jamir',
'Weber', '{\"address\": {\"street\": \"4808 Sherman Street\", \"City\": \"Lawrence\"}}',
'+1 218-980-9479');
```

```
INSERT INTO `e-commerce`.`customers` (`email`, `hashed_password`,
`customer_firstname`, `customer_lastname`, `address`, `telephone`) VALUES
('vajalod304@probdd.com',
'8b89b7de6c1c9847953d37188f651a4392c46604ee7809fbb4053d83e366af83',
'Landon', 'Mclean', '{\"address\": {\"street\": \"2626 Glenview Drive\", \"City\":
\"Corpus Christi\"}}', '+1 207-384-2684');
INSERT INTO `e-commerce`.`customers` (`email`, `hashed_password`,
`customer_firstname`, `customer_lastname`, `address`, `telephone`) VALUES
('pofaw36777@kuvasin.com',
'512c0b1ea4c1197ceb3ced3703a0c4e2581e8afb5caeb0da405b3e2eeba089d1', 'Erik',
'Finley', '{\"address\": {\"street\": \"2900 Olen Thomas Drive\", \"City\": \"Wichita
Falls\"}}', '+1 435-735-6720');
INSERT INTO `e-commerce`.`customers` (`email`, `hashed_password`,
`customer_firstname`, `customer_lastname`, `address`, `telephone`) VALUES
('sapobe7142@probdd.com',
'bc4e02087b195e22436957f331f3144da4bec1f317a21f85b1a4e165ba0b0e4f',
'Karma', 'Liu', '{\"address\": {\"street\": \"853 Woodland Terrace\", \"City\":
\"Sacramento\"}}', '+1 505-219-2420');
INSERT INTO `e-commerce`.`customers` (`email`, `hashed_password`,
`customer_firstname`, `customer_lastname`, `address`, `telephone`) VALUES
('kaxoc93466@rubeshi.com',
'8f13e0e3a432c7f5ec515fd213ce434368f2d87aca4baa1b02c871569ce1bb12',
'Shayna', 'Horn', '{\"address\": {\"street\": \"261 Oakridge Farm Lane\", \"City\":
\"Milwaukee\"}}', '+1 225-928-5202');
INSERT INTO `e-commerce`.`customers` (`email`, `hashed_password`,
`customer_firstname`, `customer_lastname`, `address`, `telephone`) VALUES
('jeyayos488@kuvasin.com',
'd4566c7ba4955b031d50fd727150d6862d2d3b947fac38688a2d384863a0ea09',
'Jeramiah', 'Villegas', '{\"address\": {\"street\": \"3419 Camden Street\", \"City\":
\"Reno\"}}', '+1 505-406-9701');
```

```
COMMIT;


-- -----------------------------------------------------
-- Data for table `e-commerce`.`cart_items`
-- -----------------------------------------------------
START TRANSACTION;
USE `e-commerce`;
INSERT INTO `e-commerce`.`cart_items` (`product_id`, `customer_id`, `quantity`)
VALUES (1, 1, 1);
INSERT INTO `e-commerce`.`cart_items` (`product_id`, `customer_id`, `quantity`)
VALUES (2, 1, 2);
INSERT INTO `e-commerce`.`cart_items` (`product_id`, `customer_id`, `quantity`)
VALUES (6, 4, 1);
INSERT INTO `e-commerce`.`cart_items` (`product_id`, `customer_id`, `quantity`)
VALUES (3, 2, 2);
INSERT INTO `e-commerce`.`cart_items` (`product_id`, `customer_id`, `quantity`)
VALUES (4, 5, 2);

COMMIT;


-- -----------------------------------------------------
-- Data for table `e-commerce`.`orders`
-- -----------------------------------------------------
START TRANSACTION;
USE `e-commerce`;
INSERT INTO `e-commerce`.`orders` (`customer_id`, `total_amount`, `status`,
`tracking_number`, `arrival_date`, `delivery_person`) VALUES (1, 2, 'Completed',
'FSW3JFFUGA', '2020-01-25', 'Leah Vincent');
INSERT INTO `e-commerce`.`orders` (`customer_id`, `total_amount`, `status`,
`tracking_number`, `arrival_date`, `delivery_person`) VALUES (2, 3, 'Completed',
'FXXJJNZQJE', '2020-01-25', 'Roberta Castaneda');
```

```
INSERT INTO `e-commerce`.`orders` (`customer_id`, `total_amount`, `status`, `tracking_number`, `arrival_date`, `delivery_person`) VALUES (6, 1, 'Awaiting Pickup', 'BYZ9XEWKA3', '2020-03-27', 'Terry Roth');
INSERT INTO `e-commerce`.`orders` (`customer_id`, `total_amount`, `status`, `tracking_number`, `arrival_date`, `delivery_person`) VALUES (3, 3, 'Awaiting Pickup', 'NGLDACCYZ7', '2021-06-05', 'Lynn Weber');
INSERT INTO `e-commerce`.`orders` (`customer_id`, `total_amount`, `status`, `tracking_number`, `arrival_date`, `delivery_person`) VALUES (4, 2, 'Awaiting Pickup', 'BXTJ94VGEM', '2021-10-07', 'Sean Dennis');
INSERT INTO `e-commerce`.`orders` (`customer_id`, `total_amount`, `status`, `tracking_number`, `arrival_date`, `delivery_person`) VALUES (1, 3, 'Declined', '36QNFLTDD5', '2021-10-20', 'Mr. Curtis Grimes');
INSERT INTO `e-commerce`.`orders` (`customer_id`, `total_amount`, `status`, `tracking_number`, `arrival_date`, `delivery_person`) VALUES (2, 4, 'Refunded', 'YAWAEN2H4S', '2022-06-27', 'Jennifer Phillips');
INSERT INTO `e-commerce`.`orders` (`customer_id`, `total_amount`, `status`, `tracking_number`, `arrival_date`, `delivery_person`) VALUES (6, 2, 'Awaiting Shipment', 'CLQ9RWNDEE', '2022-06-25', 'Laura Hall');
INSERT INTO `e-commerce`.`orders` (`customer_id`, `total_amount`, `status`, `tracking_number`, `arrival_date`, `delivery_person`) VALUES (3, 1, 'Awaiting Fulfillment', '3JHKAYFQ9K', '2022-10-30', 'David Nolan');
INSERT INTO `e-commerce`.`orders` (`customer_id`, `total_amount`, `status`, `tracking_number`, `arrival_date`, `delivery_person`) VALUES (4, 3, 'Pending', NULL, NULL, NULL);

COMMIT;


-- -----------------------------------------------------
-- Data for table `e-commerce`.`order_items`
-- -----------------------------------------------------
START TRANSACTION;
```

```
USE `e-commerce`;
INSERT INTO `e-commerce`.`order_items` (`product_id`, `order_id`, `quantity`)
VALUES (1, 1, 2);
INSERT INTO `e-commerce`.`order_items` (`product_id`, `order_id`, `quantity`)
VALUES (2, 1, 2);
INSERT INTO `e-commerce`.`order_items` (`product_id`, `order_id`, `quantity`)
VALUES (6, 4, 1);
INSERT INTO `e-commerce`.`order_items` (`product_id`, `order_id`, `quantity`)
VALUES (3, 2, 3);
INSERT INTO `e-commerce`.`order_items` (`product_id`, `order_id`, `quantity`)
VALUES (4, 5, 2);

COMMIT;


-- -----------------------------------------------------
-- Data for table `e-commerce`.`product_reviews`
-- -----------------------------------------------------
START TRANSACTION;
USE `e-commerce`;
INSERT INTO `e-commerce`.`product_reviews` (`customer_id`, `product_id`,
`rating_score`, `review`) VALUES (1, 1, 4, 'I use this in my bathroom. It\'s pretty loud
so it\'s easy to hear while your taking a shower. That said however I\'m upgrading
to the Echo Dot 4th Generation with the clock. It\'s a sphere and releasing in
October or November. The 3rd Gen Dot I have now has the clock on it.');
INSERT INTO `e-commerce`.`product_reviews` (`customer_id`, `product_id`,
`rating_score`, `review`) VALUES (2, 1, 4, 'I got one when they came out. Don\'t use
it very much but it sounds very good so it\'s nice to have on my desk for some
music here and there. But for smart home control I\'ve got Alexa devices.');
INSERT INTO `e-commerce`.`product_reviews` (`customer_id`, `product_id`,
`rating_score`, `review`) VALUES (6, 4, 5, NULL);
```

```
INSERT INTO `e-commerce`.`product_reviews` (`customer_id`, `product_id`,
`rating_score`, `review`) VALUES (3, 2, 2, NULL);
INSERT INTO `e-commerce`.`product_reviews` (`customer_id`, `product_id`,
`rating_score`, `review`) VALUES (4, 5, 3, NULL);

COMMIT;


-- -----------------------------------------------------
-- Data for table `e-commerce`.`coupons`
-- -----------------------------------------------------
START TRANSACTION;
USE `e-commerce`;
INSERT INTO `e-commerce`.`coupons` (`code`, `coupon_desc`, `discount_percent`,
`discount_amount`, `discount_limit`, `expiration_date`) VALUES ('BF2022', NULL,
0.05, NULL, 99, '2022-11-28');
INSERT INTO `e-commerce`.`coupons` (`code`, `coupon_desc`, `discount_percent`,
`discount_amount`, `discount_limit`, `expiration_date`) VALUES ('XOXO12', NULL,
NULL, 1000, NULL, '2022-09-10');
INSERT INTO `e-commerce`.`coupons` (`code`, `coupon_desc`, `discount_percent`,
`discount_amount`, `discount_limit`, `expiration_date`) VALUES ('HPNY2023', NULL,
NULL, 1500, NULL, '2023-01-31');
INSERT INTO `e-commerce`.`coupons` (`code`, `coupon_desc`, `discount_percent`,
`discount_amount`, `discount_limit`, `expiration_date`) VALUES ('SALE75', NULL,
0.75, NULL, 2000, '2023-02-25');

COMMIT;


-- -----------------------------------------------------
-- Data for table `e-commerce`.`products_coupons`
-- -----------------------------------------------------
START TRANSACTION;
```

```
USE `e-commerce`;
INSERT INTO `e-commerce`.`products_coupons` (`coupon_id`, `product_id`) VALUES
(1, 3);
INSERT INTO `e-commerce`.`products_coupons` (`coupon_id`, `product_id`) VALUES
(2, 6);
INSERT INTO `e-commerce`.`products_coupons` (`coupon_id`, `product_id`) VALUES
(3, 5);
INSERT INTO `e-commerce`.`products_coupons` (`coupon_id`, `product_id`) VALUES
(4, 2);

COMMIT;
```