```
create table city
(
  city_id int auto_increment primary key,
  city_name varchar(100) not null,
  postal varchar(20)
);

create table status_tracker
(
  status_tracker_id int auto_increment primary key,
  status_name varchar(50) not null
);

create table supplier
(
  supplier_id int auto_increment primary key,
  supplier_name varchar(100) not null,
  contact_person varchar(100),
  email varchar(255),
  phone varchar(20)
);

create table units
(
  unit_id int auto_increment primary key,
  unit_name varchar(50) not null,
  unit_quantity int not null
);

create table category
(
  category_id int auto_increment primary key,
  name varchar(100) not null,
  description text
);

create table customer
(
  customer_id int auto_increment primary key,
  customer_name varchar(100) not null,
  user_name varchar(50) not null,
  password varchar(255) not null,
  payment_profile varchar(255) not null,
  customer_email varchar(255) not null unique,
  customer_phone varchar(20)  not null,
  customer_address varchar(255) not null,
  delivery_address varchar(255),
  city_id int not null,
  delivery_city_id int,
  foreign key (city_id) references city(city_id),
  foreign key (delivery_city_id) references city(city_id)
);

create table employee
```

```
(
  employee_id int auto_increment primary key,
  employee_name varchar(100) not null,
  employee_title varchar(100) not null,
  birth_date date not null,
  state_of_residence varchar(100) not null,
  hours_worked_per_day decimal(4,2) not null
);

create table item
(
  item_id int auto_increment primary key,
  item_name varchar(100) not null,
  item_price decimal(10,2) not null check (item_price > 0),
  item_description text,
  total_sales int default 0,
  image_url varchar(255),
  unit_id int,
  foreign key (unit_id) references units(unit_id)
);

create table item_supplier
(
  item_id int not null,
  supplier_id int not null,
  unit_cost decimal(10,2) not null,
  last_delivery_date date,
  primary key (item_id, supplier_id),
  foreign key (item_id) references item(item_id),
  foreign key (supplier_id) references supplier(supplier_id)
);

create table item_category
(
  item_id int not null,
  category_id int not null,
  primary key (item_id, category_id),
  foreign key (item_id) references item(item_id),
  foreign key (category_id) references category(category_id)
);

create table inventory
(
  inventory_id int auto_increment primary key,
  quantity int not null,
  last_updated timestamp default current_timestamp on update
current_timestamp,
  item_id int not null,
  foreign key (item_id) references item(item_id)
);


create table placed_order
(
```

```
  placed_order_id int auto_increment primary key,
  customer_id int not null,
  delivery_city_id int not null,
  placed_order_date date not null,
  delivery_address varchar(255),
  total_price decimal(10,2) not null,
  foreign key (customer_id) references customer(customer_id),
  foreign key (delivery_city_id) references city(city_id)
);

create table order_status
(
  order_status_id int auto_increment primary key,
  placed_order_id int not null,
  status_tracker_id int not null,
  status_time timestamp not null,
  foreign key (placed_order_id) references
placed_order(placed_order_id),
  foreign key (status_tracker_id) references
status_tracker(status_tracker_id)
);

create table order_item (
  order_item_id int auto_increment primary key,
  quantity int not null,
  price decimal(10,2) not null,
  placed_order_id int not null,
  item_id int not null,
  foreign key (placed_order_id) references
placed_order(placed_order_id),
  foreign key (item_id) references item(item_id)
);

create table review
(
  review_id int auto_increment primary key,
  rating int not null,
  comment text,
  review_date date not null,
  item_id int not null,
  customer_id int not null,
  foreign key (item_id) references item(item_id),
  foreign key (customer_id) references customer(customer_id)
);

create table wishlist
(
  wishlist_id int auto_increment primary key,
  date_added date not null,
  item_id int not null,
  customer_id int not null,
  foreign key (item_id) references item(item_id),
  foreign key (customer_id) references customer(customer_id)
);
```

```
create table rewards
(
  rewards_id int auto_increment primary key,
  points_balance int not null,
  used_in_order_id int,
  customer_id int not null,
  foreign key (used_in_order_id) references
placed_order(placed_order_id),
  foreign key (customer_id) references customer(customer_id)
);

create table delivery
(
  delivery_id int auto_increment primary key,
  delivery_time timestamp,
  delivery_instructions text,
  scheduled_time_slot varchar(50),
  status varchar(50),
  placed_order_id int not null,
  employee_id int not null,
  foreign key (placed_order_id) references
placed_order(placed_order_id),
  foreign key (employee_id) references employee(employee_id)
);

create table buy
(
  buy_id int auto_increment primary key,
  buy_code varchar(50) not null,
  delivery_id int not null,
  employee_id int not null,
  foreign key (delivery_id) references delivery(delivery_id),
  foreign key (employee_id) references employee(employee_id)
);

create table shopping_cart
(
  cart_id int auto_increment primary key,
  quantity int not null,
  buy_id int not null,
  item_id int not null,
  foreign key (buy_id) references buy(buy_id),
  foreign key (item_id) references item(item_id)
);

create table info
(
  info_id int auto_increment primary key,
  placed_order_id int not null,
  employee_id int not null,
  customer_id int not null,
  foreign key (placed_order_id) references
placed_order(placed_order_id),
```

```
  foreign key (employee_id) references employee(employee_id),
  foreign key (customer_id) references customer(customer_id)
);


SELECT DATE(p.placed_order_date) AS sale_date,
       SUM(p.total_price) AS total_sales
FROM placed_order p
JOIN customer c ON p.customer_id = c.customer_id
WHERE DATE(p.placed_order_date) = CURDATE()
GROUP BY sale_date;


SELECT i.item_name,
       SUM(oi.quantity) AS total_sold,
       s.supplier_name
FROM item i
JOIN order_item oi ON i.item_id = oi.item_id
JOIN item_supplier isup ON i.item_id = isup.item_id
JOIN supplier s ON isup.supplier_id = s.supplier_id
GROUP BY i.item_name, s.supplier_name
ORDER BY total_sold DESC
LIMIT 5;


SELECT i.item_name,
       inv.quantity,
       c.name AS category_name
FROM item i
JOIN inventory inv ON i.item_id = inv.item_id
JOIN item_category ic ON i.item_id = ic.item_id
JOIN category c ON ic.category_id = c.category_id
ORDER BY inv.quantity ASC;



SELECT i.item_name,
       inv.quantity,
       s.supplier_name,
       isup.last_delivery_date
FROM item i
JOIN inventory inv ON i.item_id = inv.item_id
JOIN item_supplier isup ON i.item_id = isup.item_id
JOIN supplier s ON isup.supplier_id = s.supplier_id
WHERE inv.quantity = (SELECT MIN(quantity) FROM inventory)
LIMIT 1;


SELECT i.item_name,
       r.rating,
       r.comment,
       c.customer_name,
       cat.name AS category_name
FROM review r
```

```
JOIN customer c ON r.customer_id = c.customer_id
JOIN item i ON r.item_id = i.item_id
JOIN item_category ic ON i.item_id = ic.item_id
JOIN category cat ON ic.category_id = cat.category_id
ORDER BY r.review_date DESC
LIMIT 10;
```