

Customer Order Processing System

CS3462 Project



group 09

Akhil Jain (40051215)

Hadrien Luttiau (40056002)

Contents

[Introduction 3](#_Toc374035337)

[Business requirements 3](#_Toc374035338)

[System functional specifications 4](#_Toc374035339)

[Query 1 4](#_Toc374035340)

[Query 2 4](#_Toc374035341)

[Query 3 4](#_Toc374035342)

[Query 4 4](#_Toc374035343)

[Query 5 4](#_Toc374035344)

[Query 6 4](#_Toc374035345)

[Query 7 5](#_Toc374035346)

[Query 8 5](#_Toc374035347)

[Query 9 5](#_Toc374035348)

[Query 10 5](#_Toc374035349)

[Query 11 5](#_Toc374035350)

[Query 12 5](#_Toc374035351)

[Query 13 6](#_Toc374035352)

[Extended Entity Relationship Model 7](#_Toc374035353)

[Data Flow Diagrams 8](#_Toc374035354)

[Report 1 8](#_Toc374035355)

[Report 2 8](#_Toc374035356)

[Report 3 8](#_Toc374035357)

[Report 4 9](#_Toc374035358)

[Report 5 9](#_Toc374035359)

[Report 6 9](#_Toc374035360)

[Report 7 10](#_Toc374035361)

[Report 8 10](#_Toc374035362)

[Report 9 10](#_Toc374035363)

[Report 10 11](#_Toc374035364)

[Report 11 11](#_Toc374035365)

[Report 12 11](#_Toc374035366)

[Report 13 12](#_Toc374035367)

[Relational schema 13](#_Toc374035368)

[General 13](#_Toc374035369)

[Detailed 13](#_Toc374035370)

[Relational database normal form specification 16](#_Toc374035371)

[Test cases of the developed database system 17](#_Toc374035372)

[SQL programs listing 18](#_Toc374035373)

[Query 1 18](#_Toc374035374)

[Query 2 18](#_Toc374035375)

[Query 3 18](#_Toc374035376)

[Query 4 18](#_Toc374035377)

[Query 5 19](#_Toc374035378)

[Query 6 19](#_Toc374035379)

[Query 7 19](#_Toc374035380)

[Query 8 19](#_Toc374035381)

[Query 9 19](#_Toc374035382)

[Query 10 19](#_Toc374035383)

[Query 11 19](#_Toc374035384)

[Query 12 20](#_Toc374035385)

[Query 13 20](#_Toc374035386)

[ScreenCAM Demonstration CD 21](#_Toc374035387)

# Introduction

This project is to design and implement a database system for a customer order processing system in a company using a relational database Server. We have used an MS SQL based relational DBMS to implement a web based server to manage the data.

# Business requirements

The target of our relational database system is an enterprise that consists of a number of stores located in different cities and states. Each state has many cities, and a headquarter address coordinating all of its stores in the state. The location of the stores is also recorded. Each store is located in one city, and there can be many stores in the city. The enterprise’s goal is to meet all of the customer’s requirements from stores located in the customer’s city. If the requirement cannot be met, the company will turn to the other cities where the item can be found if there is any. Each store holds a variety of items in various quantity. Each item can be held in different stores. In addition, the enterprise keeps the information of the customers. The city location of the customer, together with the data of the customer’s first order, is stored by the existing system.

There are two kinds of customers: Employee and Regular, exclusively. There are also two kinds of customers on the market: Walk-in and e-commerce, inclusively. Each customer lives in one city only, and the enterprise will try to satisfy the customer’s order items by the present stock in the city where the customer lives. The order of each customer can be for any quantity of any number of items, and each order is uniquely identified by an order number. The same item can be in different orders.

Some processing information is important for the enterprise. For example, the total quantity of item stored in each city. After every time an item is taken, the company needs to know the total quantities of the item in all the stores in a city.

The retail store ordering system must produce the following reports:

1. Find all the stores along with city, state, phone, description, size, weight and unit price that hold a particular item of stock.
2. Find all the orders along with customer id and order date that can be fulfilled by a given store.
3. Find all stores along with city name and phone that hold items ordered by a given customer.
4. Find the headquarter address along with city and state of all stores that hold stocks of an item above a particular level.
5. For each customer order, show the items ordered along with description, store id and city name and the stores that hold the items.
6. Find the city and the state in which a given customer lives.
7. Find the stock level of a particular item in all stores in a particular city.
8. Find the items, quantity ordered, customer and city of an order.
9. Find a list of employee customers with name and discount rate.
10. Find a list of non-employee customers with name and post address.
11. Find a list of all customers sorted by sales volume in ascending order.
12. Find a list of walk-in customer sorted by name.
13. Find a list of ecommerce customer sorted by email address.

# System functional specifications

## Query 1

Find all the stores along with city, state, phone, description, size, weight and unit price that hold a particular item of stock.

**Input:** Item\_id

**Process:** A query that accepts input and generates an output report

**Output:** A report consists of Item\_id, Store\_id, City\_name, State, Phone, Description, Size, Weight and Unit\_price

## Query 2

Find all the orders along with customer id and order date that can be fulfilled by a given store.

**Input:** Store\_id

**Process:** A query that accepts input and generates an output report

**Output:** A report consists of Order\_no, Customer\_id and Order\_date

## Query 3

Find all stores along with city name and phone that hold items ordered by a given customer.

**Input:** Customer\_id

**Process:** A query that accepts input and generates an output report

**Output:** A report consists of Store\_id, City\_name and Phone

## Query 4

Find the headquarter address along with city and state of all stores that hold stocks of an item above a particular level.

**Input:** Item\_id, Quantity\_held

**Process:** A query accepts input and generates an output report

**Output:** A report consists of City\_name, State and Headquarter\_addr

## Query 5

For each customer order, show the items ordered along with description, store id and city name and the stores that hold the items.

**Input:** N/A

**Process:** A query that generates an output report

**Output:** A report consists of Item\_id, Description, Store\_id and City\_name associated with each order\_no.

## Query 6

Find the city and the state in which a given customer lives.

**Input:** Customer\_id

**Process:** A query that accepts input and generates an output report

**Output:** A report consists of City\_name and State

## Query 7

Find the stock level of a particular item in all stores in a particular city.

**Input:** Item\_id, City\_name

**Process:** A query that accepts input and generates an output report

**Output:** A report consists of stock level of the selected item in the selected city

## Query 8

Find the items, quantity ordered, customer and city of an order.

**Input:** Order\_no

**Process:** A query that accepts input and generates an output report

**Output:** A report consists of Item\_id, Description, Quantity\_ordered, Customer\_id and City\_name

## Query 9

Find a list of employee customers with name and discount rate.

**Input:** N/A

**Process:** A query that generates an output report

**Output:** A report consists of Employee\_customer\_name and Employee\_discount\_rate

## Query 10

Find a list of non-employee customers with name and post address.

**Input:** N/A

**Process:** A query that generates an output report

**Output:** A report consists of Regular\_customer\_name and Regular\_customer\_address

## Query 11

Find a list of all customers sorted by sales volume in ascending order.

**Input:** N/A

**Process:** A query that generates an output report

**Output:** A report consists of Customer Name and his Sales Volume

## Query 12

Find a list of walk-in customer sorted by name.

**Input:** N/A

**Process:** A query that generates an output report

**Output:** A report consists of Walkin\_customer\_name

## Query 13

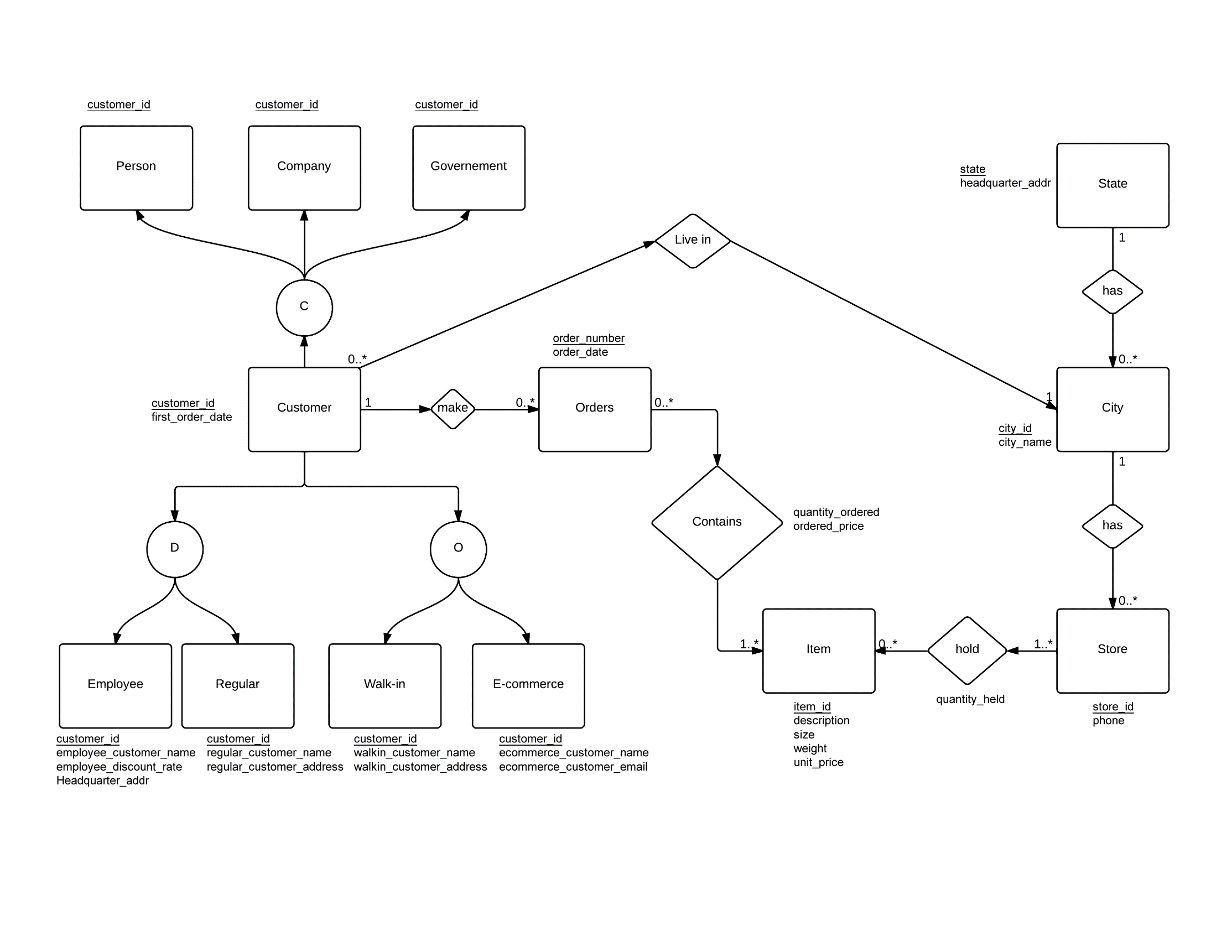
Find a list of ecommerce customer sorted by email address.

**Input:** N/A

**Process:** A query that generates an output report

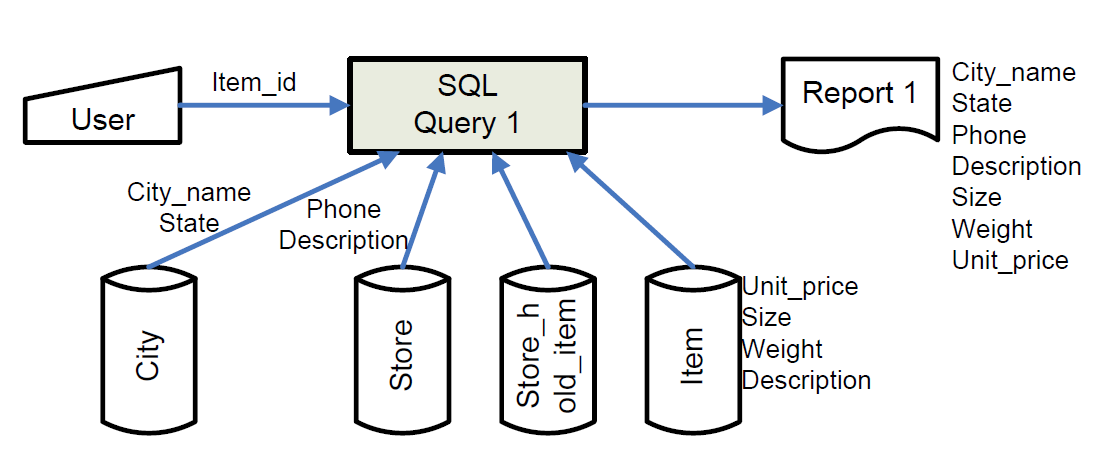
**Output:** A report consists of Ecommerce\_customer\_name

# Extended Entity Relationship Model

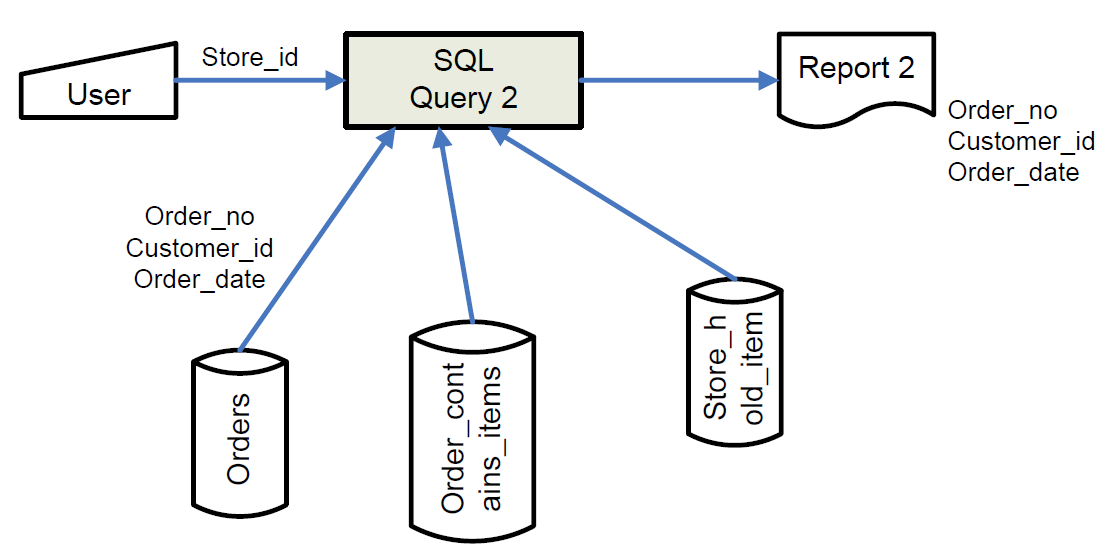


# Data Flow Diagrams

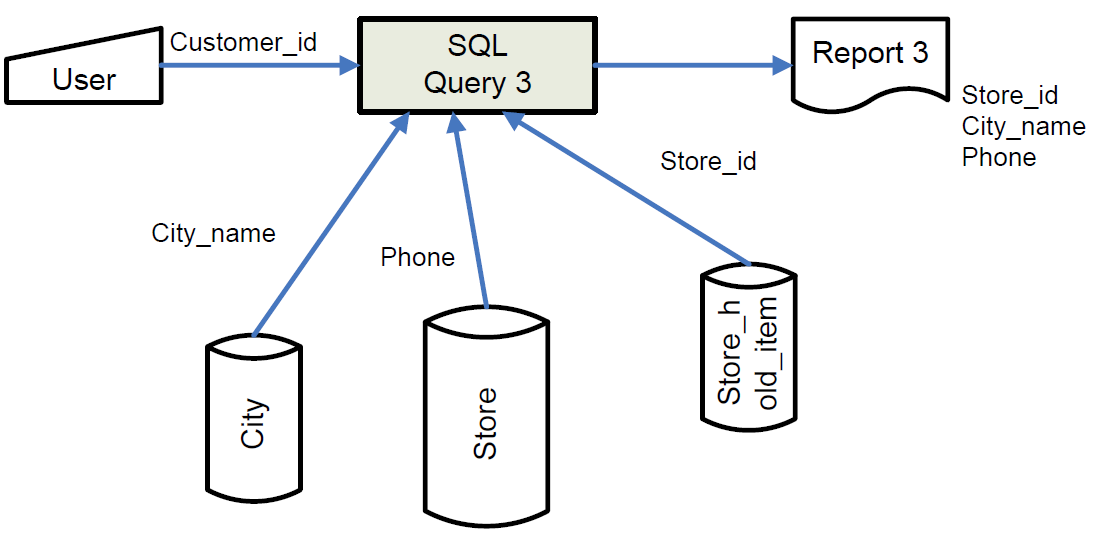
## Report 1



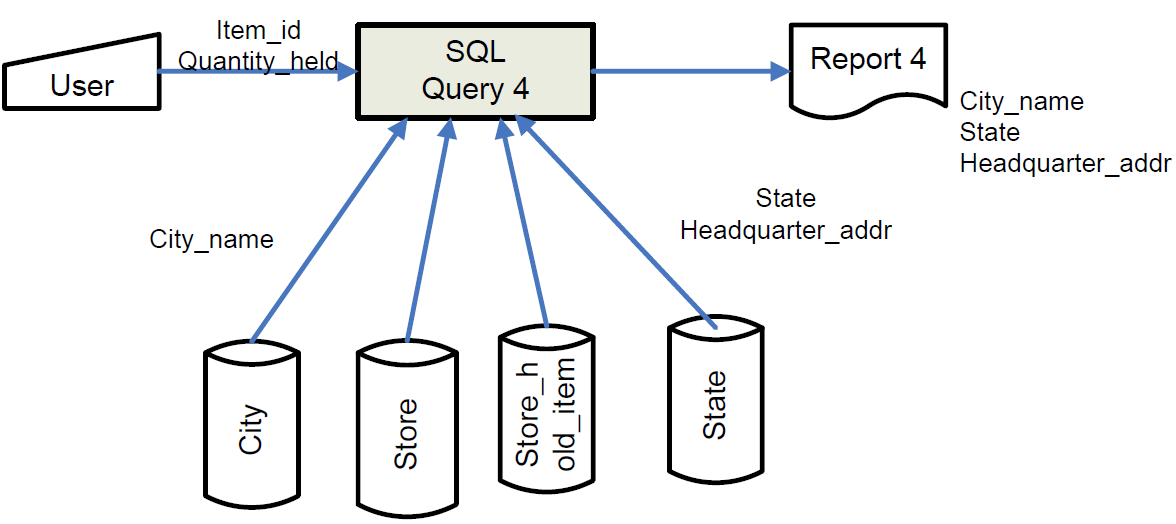
## Report 2



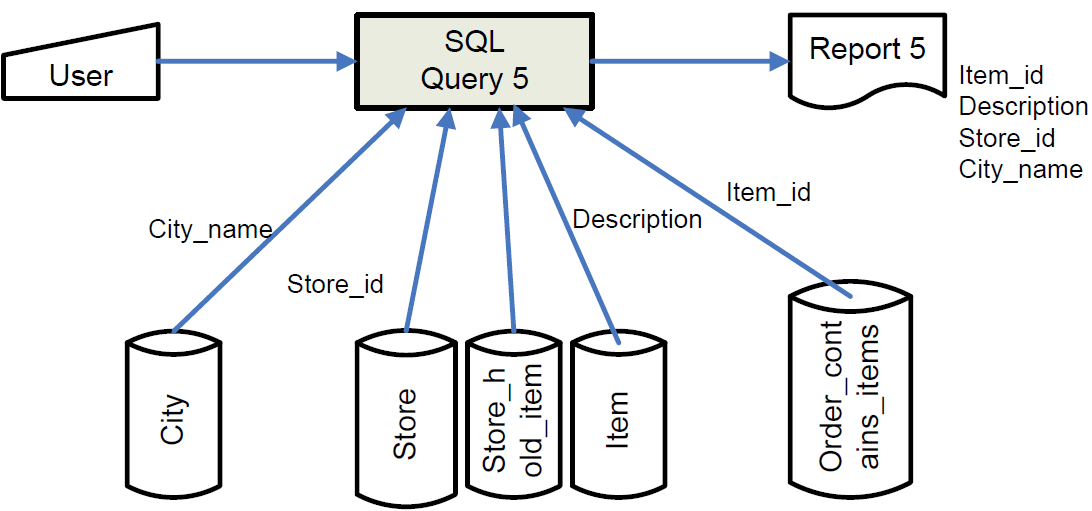
Report 3



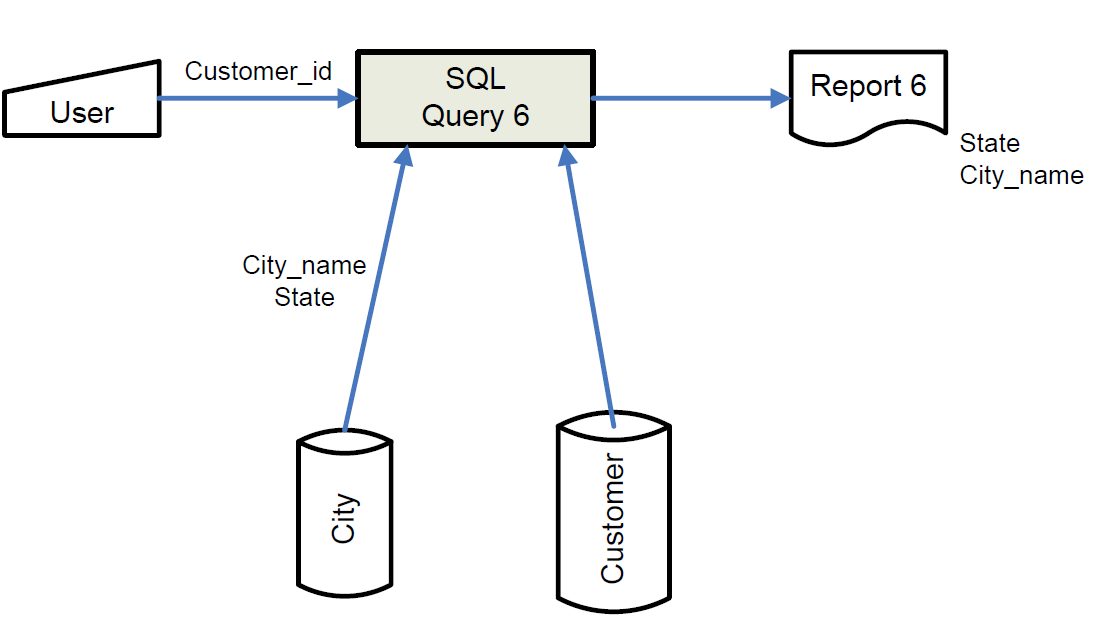
Report 4



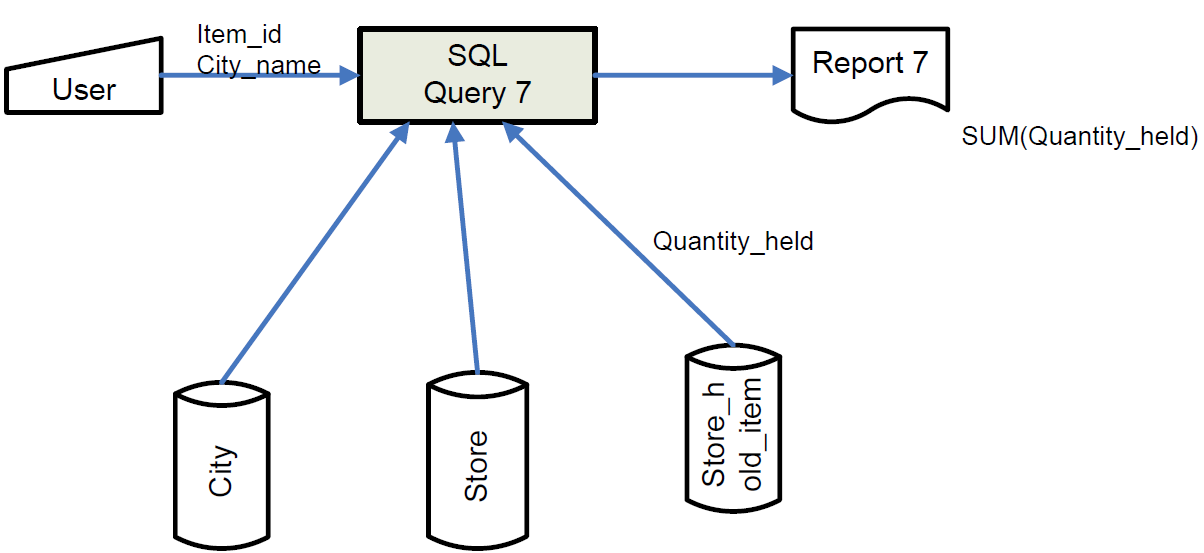
Report 5



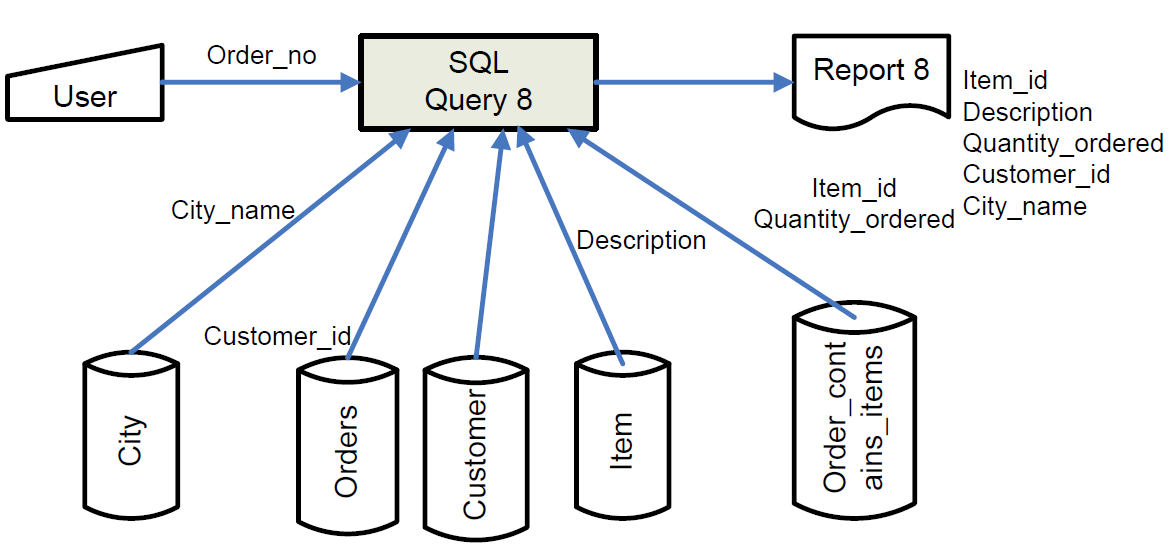
Report 6



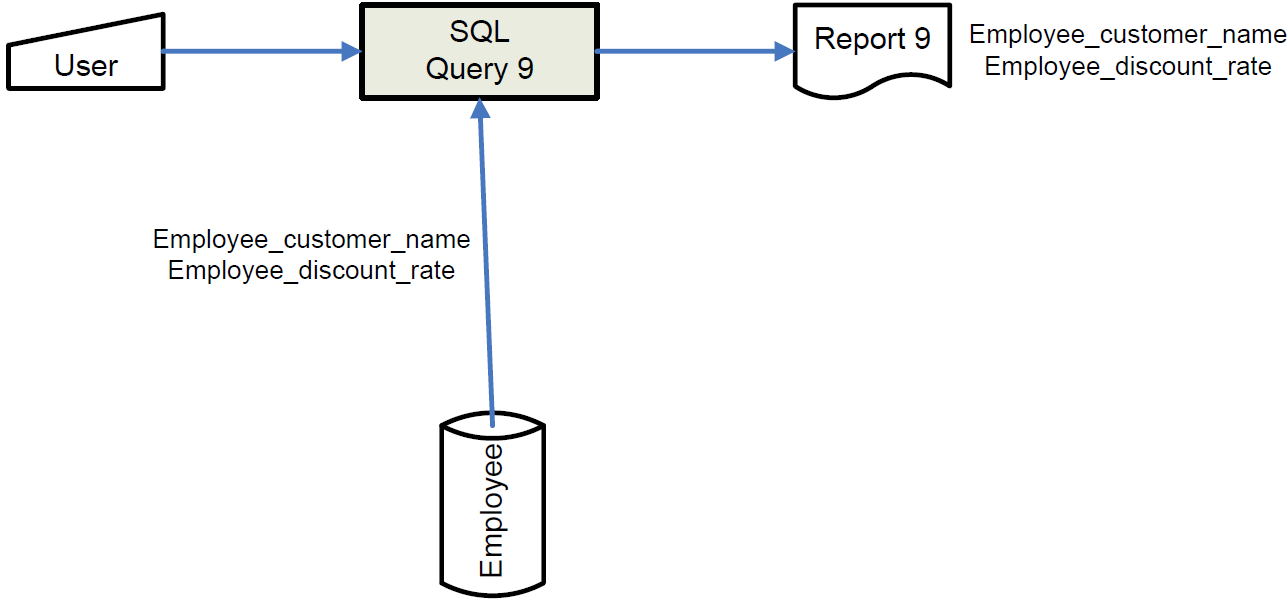
Report 7



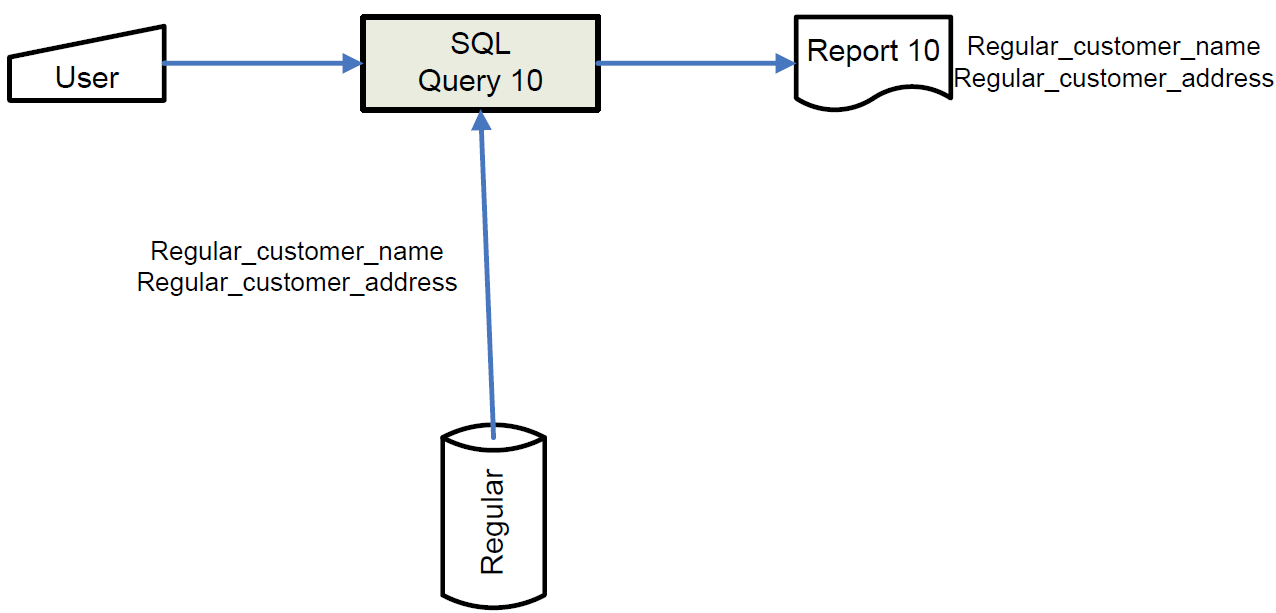
Report 8



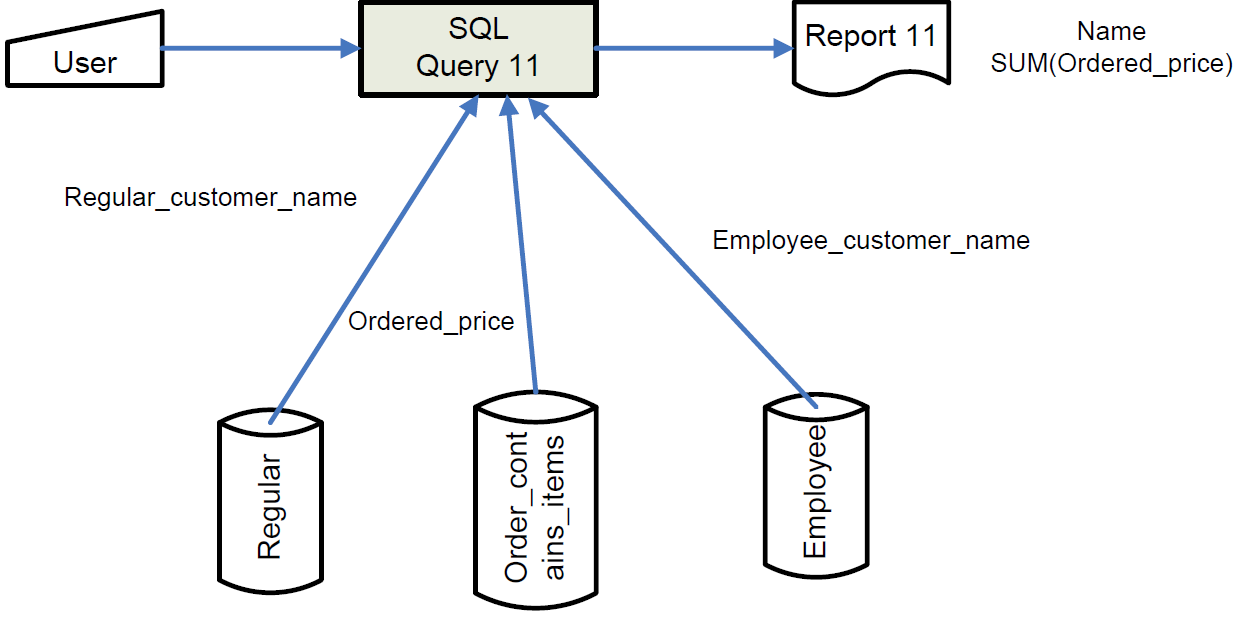
Report 9



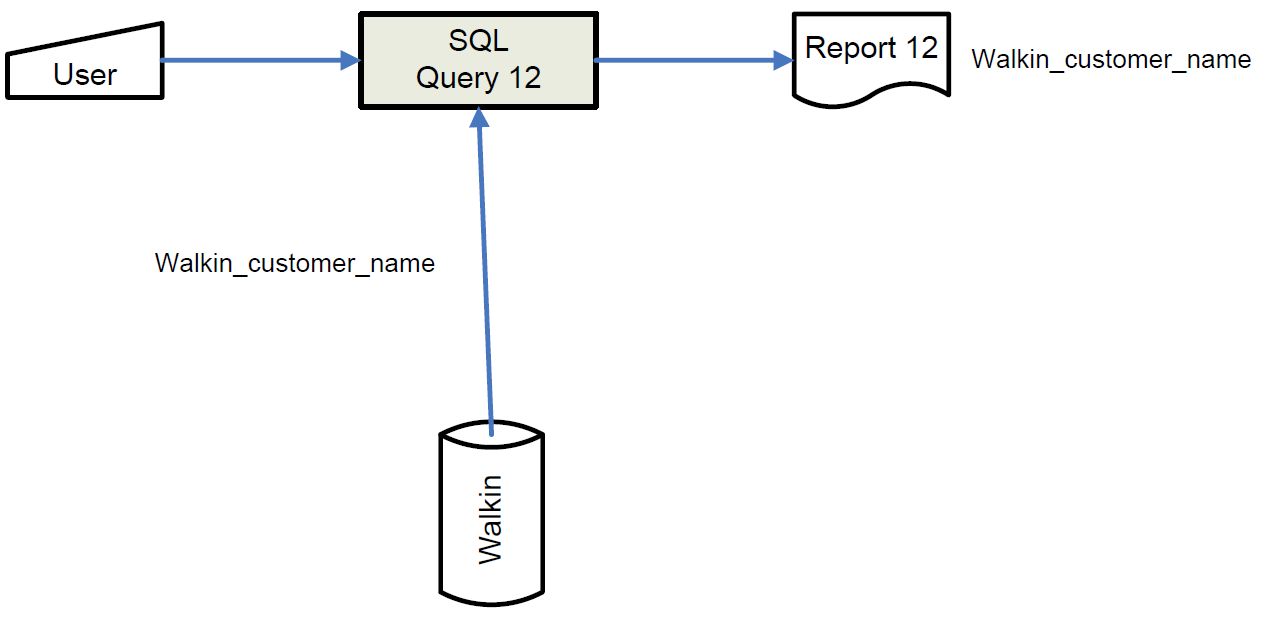
Report 10



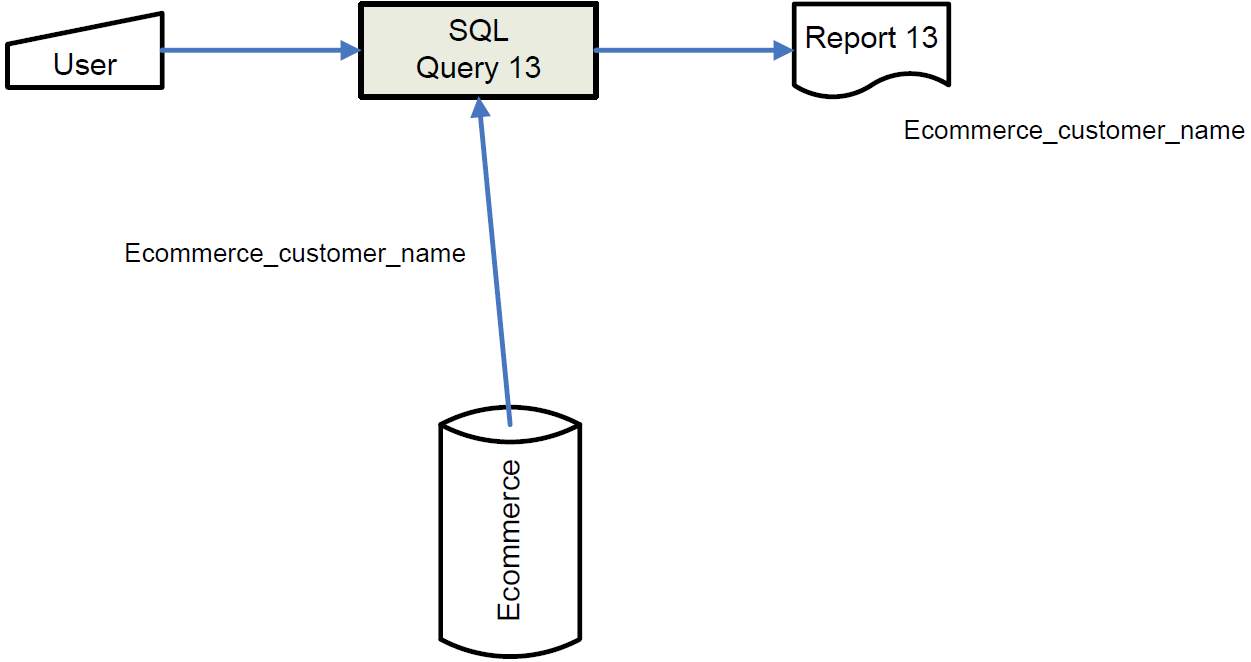
Report 11



Report 12



Report 13



# Relational schema

## General

Relation State (State, Headquarter\_addr);

Relation City (City\_id, City\_name,\*State);

Relation Store (Store\_id,\*City\_id, Phone);

Relation Item (Item\_id, Description, Size, Weight, Unit\_price);

Relation Store\_hold\_Item (\*Store\_id, \*Item\_id, Quantity\_held);

Relation Customer (Customer\_id, First\_order\_date, \*City\_id);

Relation Order (Order\_no, Order\_date, \*Customer\_id);

Relation Order\_contains\_items (\*Item\_id, \*Order\_no, Quantity\_ordered, Ordered\_price);

Relation Person (Customer\_id);

Relation Company (Customer\_id);

Relation Government (Customer\_id);

Relation Employee (\*Customer\_id, Employee\_customer\_name, Employee\_discount\_rate, Headquarter\_addr);

Relation Regular (\*Customer\_id, Regular\_customer\_name, Regular\_customer\_address);

Relation WalkIn (\*Customer\_id, Walkin\_customer\_name, Walkin\_customer\_address);

Relation ECommerce (\*Customer\_id, Ecommerce\_customer\_name, Ecommerce\_customer\_email)

## Detailed

CREATE TABLE State (

State CHAR(20),

Headquarter\_addr CHAR(4),

PRIMARY KEY(State)

);

CREATE TABLE City (

City\_id CHAR(4),

City\_name CHAR(15),

State CHAR(15),

PRIMARY KEY(City\_id),

FOREIGN KEY(State) REFERENCES States(State)

);

CREATE TABLE Store (

Store\_id CHAR(4),

City\_id CHAR(4),

Phone CHAR(8),

PRIMARY KEY(Store\_id),

FOREIGN KEY(City\_id) REFERENCES City(City\_id)

);

CREATE TABLE Item (

Item\_id CHAR(4),

Description CHAR(40),

Size CHAR(40),

Weight REAL,

Unit\_price REAL,

PRIMARY KEY(Item\_id)

);

CREATE TABLE Store\_hold\_Item (

Store\_id CHAR(4),

Item\_id CHAR(4),

Quantity\_held INT,

PRIMARY KEY(Store\_id, Item\_id),

FOREIGN KEY(Store\_id) REFERENCES Store(Store\_id),

FOREIGN KEY (Item\_id) REFERENCES Item(Item\_id)

);

CREATE TABLE Customer (

Customer\_id CHAR(4),

First\_order\_date DATE,

City\_id CHAR(4),

PRIMARY KEY(Customer\_id),

FOREIGN KEY(City\_id) REFERENCES City(City\_id)

);

CREATE TABLE Order (

Order\_no CHAR(4),

Order\_date DATE,

Customer\_id CHAR(4),

PRIMARY KEY(Order\_no),

FOREIGN KEY(Customer\_id) REFERENCES Customer(Customer\_id)

);

CREATE TABLE Order\_contains\_items (

Item\_id CHAR(4),

Order\_no INT,

Quantity\_ordered INT,

Ordered\_price REAL,

PRIMARY KEY(Item\_id, Order\_no),

FOREIGN KEY(Order\_no) REFERENCES Order(Order\_no),

FOREIGN KEY(Item\_id) REFERENCES Item(Item\_id)

);

CREATE TABLE Person (

Customer\_id CHAR(4),

PRIMARY KEY(Customer\_id)

);

CREATE TABLE Company (

Customer\_id CHAR(4),

PRIMARY KEY(Customer\_id)

);

CREATE TABLE Government (

Customer\_id CHAR(4),

PRIMARY KEY(Customer\_id)

);

CREATE TABLE Employee (

Customer\_id CHAR(4),

Employee\_customer\_name CHAR(15),

Employee\_discount\_rate FLOAT,

Headquarter\_addr CHAR(4),

PRIMARY KEY(Customer\_id)

FOREIGN KEY(Customer\_id) REFERENCES Customer(Customer\_id)

);

CREATE TABLE Regular (

Customer\_id CHAR(4),

Regular\_customer\_name CHAR(15),

Regular\_customer\_address CHAR(80),

PRIMARY KEY(Customer\_id)

FOREIGN KEY(Customer\_id) REFERENCES Customer(Customer\_id)

);

CREATE TABLE WalkIn (

Customer\_id CHAR(4),

Walkin\_customer\_name CHAR(15),

Walkin\_customer\_address CHAR(80),

PRIMARY KEY(Customer\_id)

FOREIGN KEY(Customer\_id) REFERENCES Customer(Customer\_id)

);

CREATE TABLE ECommerce (

Customer\_id CHAR(4),

Ecommerce\_customer\_name CHAR(15),

Ecommerce\_customer\_email CHAR(80),

PRIMARY KEY(Customer\_id)

FOREIGN KEY(Customer\_id) REFERENCES Customer(Customer\_id)

);

# Relational database normal form specification

**First Normal form (1NF):** The domain of each attribute contains only atomic values, and the value of each attribute contains only a single value from that domain.

**Second Normal form (2NF):** No non-prime attribute in the table is functionally dependent on a proper subset of any candidate key.

**Third Normal form (3NF):** Every non-prime attribute is non-transitively dependent on every candidate key in the table. The attributes that do not contribute to the description of the primary key are removed from the table. In other words, no transitive dependency is allowed.

**Boyce Codd Normal form (BCNF):** Every non-trivial functional dependency in the table is a dependency on a superkey.

**Fourth Normal form (4NF):** Every non-trivial multivalued dependency in the table is a dependency on a superkey.

**Fifth Normal form (5NF):** Every non-trivial join dependency in the table is implied by the superkeys of the table.

In keeping with these definitions, we can say that the relations in the developed database are normalised as follows:

|  |  |  |
| --- | --- | --- |
| Sr. No. | Relation | Normal Form |
| 1 | State | 5NF |
| 2 | City | 3NF |
| 3 | Store | 3NF |
| 4 | Item | 5NF |
| 5 | Store\_hold\_item | 3NF |
| 6 | Customer | 5NF |
| 7 | Order | 3NF |
| 8 | Order\_contains\_items | 3NF |
| 9 | Person | 5NF |
| 10 | Company | 5NF |
| 11 | Government | 5NF |
| 12 | Employee | 5NF |
| 13 | Regular | 5NF |
| 14 | Walkin | 5NF |
| 15 | Ecommerce | 5NF |

# Test cases of the developed database system

# SQL programs listing

## Query 1

SELECT item.item\_id as a, store.store\_id as b, city.city\_id as c, city.city\_name as d, city.state as e, store.phone as f, item.description as g,item.size as h, item.weight as i, item.unit\_price as j

FROM city, store, item, store\_hold\_item

WHERE city.city\_id = store.city\_id

AND store.store\_id = store\_hold\_item.store\_id

AND store\_hold\_item.item\_id = item.item\_id

AND item.item\_id =<user\_input>

ORDER BY store.store\_id;

## Query 2

SELECT Orders.order\_no as order, Orders.Customer\_id as id, Orders.order\_date as orderdate

FROM orders, Order\_contains\_items, Store\_hold\_Item

WHERE Store\_hold\_Item.store\_id=<user\_input>

AND Order\_contains\_items.order\_no=orders.order\_no

AND NOT EXISTS (

SELECT \*

FROM Order\_contains\_items, Store\_hold\_Item

WHERE Order\_contains\_items.quantity\_ordered>Store\_hold\_Item.quantity\_held

AND Store\_hold\_Item.store\_id='"+a+"'

AND Order\_contains\_items.order\_no=orders.order\_no

);

## Query 3

SELECT Store\_hold\_Item.item\_id as items, Store\_hold\_Item.store\_id as stores, city.city\_name as city, store.phone as phone

FROM Store\_hold\_Item, store, city

WHERE (

Store\_hold\_Item.item\_id

IN (

SELECT Order\_contains\_items.item\_id

FROM Order\_contains\_items

WHERE Order\_contains\_items.order\_no = Orders.order\_no

AND Orders.Customer\_id =<user\_input>

)

)

AND Store\_hold\_Item.store\_id=store.store\_id

AND store.city\_id=city.city\_id;

## Query 4

SELECT city.city\_name as city, state.state as state, state.headquarter\_addr as hq

FROM city, state, Store\_hold\_Item, store

WHERE Store\_hold\_Item.item\_id=<user\_input>

AND Store\_hold\_Item.quantity\_held><user\_input>

AND Store\_hold\_Item.store\_id=store.store\_id

AND store.city\_id=city.city\_id

AND city.state=state.state;

## Query 5

SELECT Order\_contains\_items.item\_id as id, item.description as des, store.store\_id as store, city.city\_name as city

FROM Order\_contains\_items, item, store, city, Store\_hold\_Item

WHERE Order\_contains\_items.order\_no=<cycle\_through\_orders>

AND Order\_contains\_items.item\_id=item.item\_id

AND Store\_hold\_Item.item\_id=item.item\_id

AND Store\_hold\_Item.store\_id=store.store\_id

AND store.city\_id=city.city\_id;

## Query 6

SELECT city.city\_name as city, city.state as state

FROM city, customer

WHERE Customer.Customer\_id=<user\_input>

AND Customer.city\_id=city.city\_id;

## Query 7

SELECT SUM(Store\_hold\_Item.quantity\_held) as stock, city.city\_name as city, item.item\_id as id, item.description as des

FROM Store\_hold\_Item, city, store

WHERE Store\_hold\_Item.item\_id=<user\_input>

AND store.store\_id=Store\_hold\_Item.store\_id

AND city.city\_id=store.city\_id

AND city.city\_name=<user\_input>;

## Query 8

SELECT Order\_contains\_items.item\_id as item, item.description as des, Order\_contains\_items.quantity\_ordered as qo, Orders.Customer\_id as customer, city.city\_name as city

FROM orders, item, Order\_contains\_items, customer, city

WHERE Order\_contains\_items.order\_no=<user\_input>

AND Order\_contains\_items.order\_no=orders.order\_no

AND item.item\_id=Order\_contains\_items.item\_id

AND Orders.Customer\_id=Customer.Customer\_id

AND Customer.city\_id=city.city\_name;

## Query 9

SELECT employee.Employee\_customer\_name as name, employee.Employee\_discount\_rate as dis

FROM employee

ORDER BY employee.Customer\_id;

## Query 10

SELECT regular.Regular\_customer\_name as name, regular.Regular\_customer\_address as address

FROM regular

ORDER BY regular.Customer\_id;

## Query 11

SELECT Regular.Regular\_customer\_name as name, SUM(Order\_contains\_items.ordered\_price) as sales

FROM regular, Order\_contains\_items

WHERE regular.Customer\_id=orders.Customer\_id

AND Order\_contains\_items.order\_no=orders.order\_no

UNION

SELECT Employee.Employee\_customer\_name as name, SUM(Order\_contains\_items.ordered\_price) as sales

FROM employee, Order\_contains\_items

WHERE employee.Customer\_id=orders.Customer\_id

AND Order\_contains\_items.order\_no=orders.order\_no

ORDER BY sales;

## Query 12

SELECT walkin.Walkin\_customer\_name as name

FROM walkin

ORDER BY walkin.Walkin\_customer\_name;

## Query 13

SELECT ecommerce.Ecommerce\_customer\_name as name, ecommerce.Ecommerce\_customer\_email as email

FROM ecommerce

ORDER BY ecommerce.Ecommerce\_customer\_email;

# ScreenCAM Demonstration CD