Group Project (2 person)

TIS1101

(Database Fundamentals)

Food Ordering System

Table of Contents

[Project Scope 1](#_Toc505537835)

[Business Rules 1](#_Toc505537836)

[Entity-Relationship Diagram (ERD) 2](#_Toc505537837)

[Data Dictionary 3](#_Toc505537838)

[Table creation query 4](#_Toc505537839)

[Aggregate function (Syntax) 8](#_Toc505537840)

[**Aggregate function (Screenshots)** 8](#_Toc505537841)

[**Aggregate function (Explanation)** 9](#_Toc505537842)

[Group by and having clause query (Syntax) 9](#_Toc505537843)

[**Group by and having clause query (Screenshots)** 9](#_Toc505537844)

[**Group by and having clause query (Explanation)** 10](#_Toc505537845)

[Trigger 1 (Syntax) 11](#_Toc505537846)

[**Trigger 1 (Screenshots)** 11](#_Toc505537847)

[**Trigger 1 (Explanation)** 11](#_Toc505537848)

[Trigger 2 (Syntax) 12](#_Toc505537849)

[**Trigger 2 (Screenshots)** 12](#_Toc505537850)

[**Trigger 2 (Explanation)** 13](#_Toc505537851)

[Trigger 3 (Syntax) 13](#_Toc505537852)

[**Trigger 3 (Screenshots)** 13](#_Toc505537853)

[**Trigger 3 (Explanation)** 14](#_Toc505537854)

[Stored Procedure (Syntax) 15](#_Toc505537855)

[**Stored Procedure (Screenshots)** 15](#_Toc505537856)

[**Stored Procedure (Explanation)** 16](#_Toc505537857)

[View (Syntax) 17](#_Toc505537858)

[**View (Screenshots)** 17](#_Toc505537859)

[**View (Explanation)** 17](#_Toc505537860)

[Sub queries (Syntax) 18](#_Toc505537861)

[**Sub queries (Screenshots)** 18](#_Toc505537862)

[**Sub queries (Explanation)** 19](#_Toc505537863)

[Query not covered 1 (Syntax) 20](#_Toc505537864)

[**Query not covered 1 (Screenshots)** 20](#_Toc505537865)

[**Query not covered 1 (Explanation)** 21](#_Toc505537866)

[Query not covered 2 (Syntax) 22](#_Toc505537867)

[**Query not covered 2 (Screenshots)** 22](#_Toc505537868)

[**Query not covered 2 (Explanation)** 23](#_Toc505537869)

[Value Insertion Query (Customer) 24](#_Toc505537870)

[Insert Query (Employee) 25](#_Toc505537871)

[Insert Query (Cuisine\_Type) 26](#_Toc505537872)

[Insert Query (Menu) 27](#_Toc505537873)

[Insert Query (Product\_Info) 28](#_Toc505537874)

[Insert query (Invoice) 30](#_Toc505537875)

[Insert Query (Invoice\_Details) 31](#_Toc505537876)

# Project Scope

This database is designed for food ordering purposes. It will assist the administrator in the restaurant to keep track and analyze valuable data that will play a part the restaurant’s business decision making process. The idea behind this is that this is a multi-cuisine type restaurant. The customers will enter the restaurant and order their food from a tablet on their table. An employee will bring the food out once it is ready to be served and all valuable data are kept in this database.

# Business Rules

1. A customer appears in one or many invoice.
2. An employee may generate many invoices.
3. An invoice has many invoice details.
4. A product info may appear in many invoice details.
5. A menu has many product details.
6. A cuisine type belongs to many Menu item.

# Entity-Relationship Diagram (ERD)



# Data Dictionary

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| TABLE NAME | ATTRIBUTE NAME | CONTENTS | TYPE | FORMAT | PK OR FK | FK REFERENCED TABLE |
| CUSTOMER | CUST\_ID  CUST\_NAME  CUST\_PHONE  CUST\_STATE  CUST\_CITY  CUST\_STREET  CUST\_POSCODE | Customer’s ID  Customer’s Name  Customer’s Phone  Customer’s Address(State)  Customer’s Address(City)  Customer’s Address(Street name)  Customer’s Address(Poscode) | VARCHAR(10)  VARCHAR(40)  VARCHAR(15)  VARCHAR(20)  VARCHAR(20)  VARCHAR(50)  VARCHAR(10) | CUSTxxxx  Xxxxxxxx  Xxxxxxxx  Xxxxxxxx  Xxxxxxxx  Xxxxxxxx  Xxxxxxxx | PK |  |
| EMPLOYEE | EMP\_ID  EMP\_NAME  EMP\_POSITION | Employee’s ID  Employee’s name  Employee’s rank | VARCHAR(10)  VARCHAR(40)  VARCHAR(20) | EMPxxxx  Xxxxxxxx  Xxxxxxxx | PK |  |
| INVOICE | INVOICE\_ID  INVOICE\_DATE  INVOICE\_TOTALPRICE  TABLE\_NO  EMP\_ID  CUST\_ID | Invoice’s ID  Invoice’s date  Invoice’s total price  Table number  Employee’s ID  Customer’s ID | VARCHAR(10)  DATE  DECIMAL(7,2)  INT  VARCHAR(10)  VARCHAR(10) | INVxxxx  yyyy-mm-dd  99999.99  99999  EMPxxxx  CUSTxxxx | PK  FK  FK | EMPLOYEE  CUSTOMER |
| INVOICE\_DETAILS | DETAILS\_ID  DETAILS\_PRICE  DETAILS\_QTY  INVOICE\_ID  PROD\_INFO\_ID | Details’ ID  Details’ Price  Details’ Quantity  Invoice’s ID  Product Info’s ID | VARCHAR(10)  DECIMAL(5,2)  INT  VARCHAR(10)  VARCHAR(10) | DETxxxx  999.99  99999  INVxxxx  PIxxxx | PK  FK  FK | INVOICE  PRODUCT\_INFO |
| PRODUCT\_INFO | PROD\_INFO\_ID  PROD\_INFO\_SIZE  PROD\_INFO\_PRICE  PROD\_ID | Product Info’s ID  Product Info’s size  Product Info’s Price  Product’s ID | VARCHAR(10)  CHAR(1)  DECIMAL(5,2)  VARCHAR(10) | PIxxxx  X  999.99  Pxxxx | PK  FK | MENU |
| MENU | PROD\_ID  PROD\_NAME  PROD\_TYPE  CTYPE\_ID | Product’s ID  Product Name  Product Type  Cuisine Type | VARCHAR(10)  VARCHAR(40)  VARCHAR(20)  VARCHAR(10) | Pxxxx  Xxxxxxx  Xxxxxxx  CTxxxx | PK  FK | CUISINE\_TYPE |
| CUISINE\_TYPE | CTYPE\_ID  CTYPE\_NAME | Cuisine Type ID  Cuisine Type Name | VARCHAR(10)  VARCHAR(20) | CTxxxx  Xxxxxx | PK |  |

# Table creation query

Create database asgn;

Connect to asgn;

Create table Customer

(

Cust\_ID varchar(10) not null,

Cust\_Name varchar(40),

Cust\_Phone varchar(15),

Cust\_State varchar(20),

Cust\_City varchar(20),

Cust\_Street varchar(50),

Cust\_Poscode varchar(10),

Primary key(Cust\_ID)

);

Create table Employee

(

Emp\_ID varchar(10) not null,

Emp\_Name varchar(40),

Emp\_Position varchar(20),

Primary key(Emp\_ID)

);

Create table Invoice

(

Invoice\_ID varchar(10) not null,

Invoice\_Date date,

Invoice\_TotalPrice decimal(7,2),

Table\_no int,

Emp\_ID varchar(10),

Cust\_ID varchar(10),

Primary key(Invoice\_ID),

Foreign key(Emp\_ID) references Employee,

Foreign key(Cust\_ID) references Customer

);

Create table Cuisine\_Type

(

CType\_ID varchar(10) not null,

CTypeName varchar(20),

Primary key(CType\_ID)

);

Create table Menu

(

Prod\_ID varchar(10) not null,

Prod\_Name varchar(40),

Prod\_Type varchar(20),

CType\_ID varchar(10),

Primary key(Prod\_ID),

Foreign key(CType\_ID) references Cuisine\_Type

);

Create table Product\_Info

(

Prod\_Info\_ID varchar(10) not null,

Prod\_Info\_Size char(1),

Prod\_Info\_Price decimal(5,2),

Prod\_ID varchar(10),

Primary key(Prod\_Info\_ID),

Foreign key(Prod\_ID) references Menu

);

Create table Invoice\_Details

(

Details\_ID varchar(10) not null,

Details\_Price decimal (7,2),

Details\_Qty int,

Invoice\_ID varchar(10),

Prod\_Info\_ID varchar(10),

Primary key(Details\_ID),

Foreign key(Invoice\_ID) references Invoice,

Foreign key(Prod\_Info\_ID) references Product\_Info

);

# Aggregate function (Syntax)

Create trigger inv\_totalPriceIns

after insert on Invoice\_Details

referencing new as n

for each row mode db2sql

update Invoice

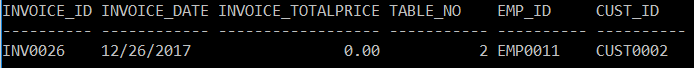
set Invoice\_TotalPrice = (select sum(Details\_Price) from Invoice\_Details

where Invoice\_Details.Invoice\_ID= n.Invoice\_ID)

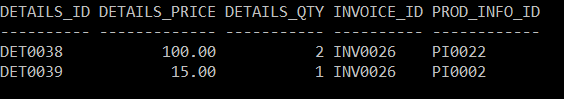
where Invoice.Invoice\_ID=n.Invoice\_ID

## **Aggregate function (Screenshots)**

Before insert into Invoice\_Details

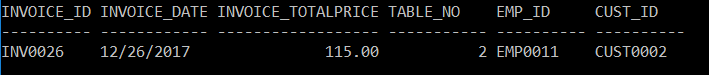


After insert query into Invoice\_Details



In Invoice with ID ‘INV0026’ there are two items, one item of quantity 2 with final price of 100.00 and another item of quantity 1 with final price of 15.00

The resulting total price in Invoice table for invoice ‘INV0026’ is the sum of both prices of the same invoice ID:



## **Aggregate function (Explanation)**

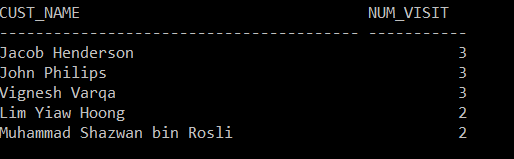
This is a trigger with aggregate function. This trigger will trigger after a new record is inserted into Invoice\_details whereby it update the Invoice\_TotalPrice in Invoice to ensure the total price of that particular invoice updates along when new record is inserted into Invoice\_Details.

# Group by and having clause query (Syntax)

Create view regularcust(Cust\_Name,Num\_Visit) as select cust\_name,count(\*) from Invoice, Customer where Invoice.Cust\_ID=Customer.Cust\_ID group by cust\_Name having count(\*)>5

Select \* from regularcust order by num\_visit desc

## **Group by and having clause query (Screenshots)**



## **Group by and having clause query (Explanation)**

This is a view with ‘group by’ and ‘having’ clause implemented. This view consist of two columns Customer’s name and his/her number of visit. The view will group by Customer’s name and will only display records with more than 5 visits to the premise.

# Trigger 1 (Syntax)

create trigger get\_detailsPrice

after insert on Invoice\_Details

referencing new as n

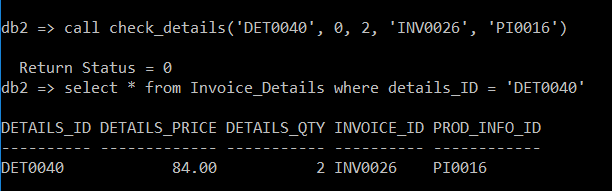
for each row mode db2sql

update Invoice\_Details

set Details\_Price = (select Prod\_Info\_Price\*Details\_Qty from Product\_Info,Invoice\_Details where Product\_Info.prod\_info\_ID=Invoice\_details.prod\_info\_ID and Invoice\_details.details\_id=n.details\_id )

where Invoice\_Details.Details\_ID= n.Details\_ID

## **Trigger 1 (Screenshots)**



## **Trigger 1 (Explanation)**

The trigger get\_detailsPrice will trigger after a new record is inserted into Invoice\_details. The trigger will calculate the details’ price according to the value of the record entered. User may opt not to insert any value for details\_price when they insert record into Invoice\_Details as the details\_price will automatically calculated by this trigger.

# Trigger 2 (Syntax)

create trigger get\_detailsPriceUpd

after update of Details\_qty on Invoice\_Details

referencing new as n

for each row mode db2sql

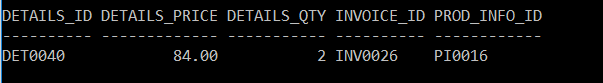
update Invoice\_Details

set Details\_Price = (select Prod\_Info\_Price\*Details\_Qty from Product\_Info,Invoice\_Details where Product\_Info.prod\_info\_ID=Invoice\_details.prod\_info\_ID and Invoice\_details.details\_id=n.details\_id )

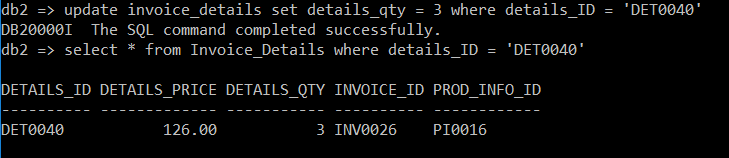
where Invoice\_Details.Details\_ID= n.Details\_ID

## **Trigger 2 (Screenshots)**

Before update



After update on Details\_QTY



## **Trigger 2 (Explanation)**

The trigger get\_detailsPrice will trigger after a details\_qty is updated into Invoice\_details. As an example user updated the quantity of the item (Details\_Qty) which will affect the Details\_Price to change. This trigger will recalculate the details’ price according to the updated details\_qty.

# Trigger 3 (Syntax)

Create trigger inv\_totalPriceUpd

after update on Invoice\_Details

referencing new as n

for each row mode db2sql

update Invoice

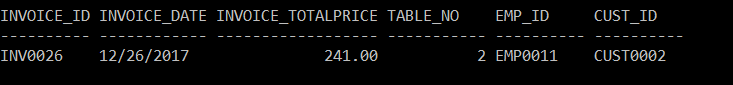
set Invoice\_TotalPrice = (select sum(Details\_Price) from Invoice\_Details

where Invoice\_Details.Invoice\_ID= n.Invoice\_ID)

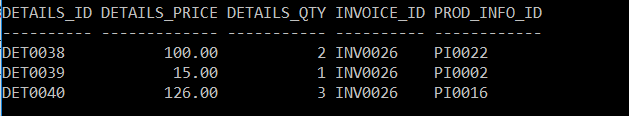
where Invoice.Invoice\_ID=n.Invoice\_ID

## **Trigger 3 (Screenshots)**

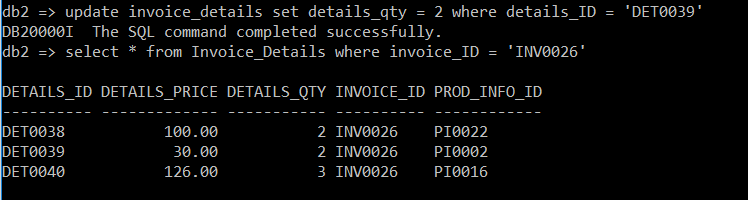
Before update on Invoice\_Details in Invoice table:



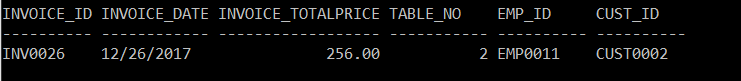
Before update on Invoice\_Details in Invoice\_Details table:



After update on Invoice\_Details in Invoice\_Details table:



After update on Invoice\_Details in Invoice table:



## **Trigger 3 (Explanation)**

The trigger inv\_totalPriceUPD will trigger after any update in Invoice\_details. As an example the user updated the quantity of item (Details\_Qty). This will cause the total price in Invoice table to change hence this trigger will recalculate the Invoice\_TotalPrice in Invoice.

# Stored Procedure (Syntax)

Create procedure check\_details(in dID varchar(10),price decimal(5,2),Qty int,iID varchar(10),

Pid varchar(10))

begin

if pID not in(select prod\_info\_id from invoice\_details where invoice\_details.Invoice\_ID=iID) then

insert into Invoice\_details values(dID,Price,Qty,iID,pID);

else

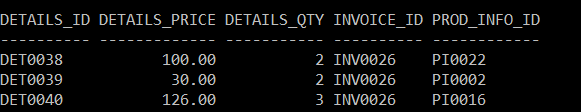
Signal sqlstate ‘75001’ set message\_text=’Cannot enter existing products into same invoices’;

end if;

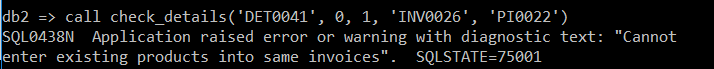
end

## **Stored Procedure (Screenshots)**

Current Invoice\_Details values:



After trying to insert an Invoice details of the same invoice with an existing item:



## **Stored Procedure (Explanation)**

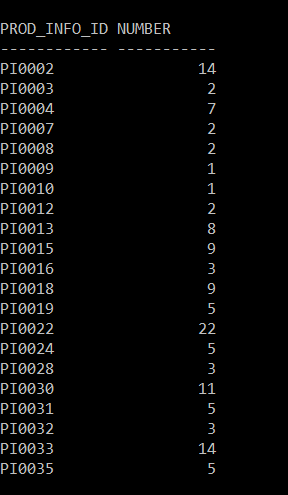
This stored procedure is used to insert values into Invoice\_Details. This is an error checking stored procedure where it checks if the Prod\_Info\_ID entered has already exist in the same invoice. If it already exist, the user will be informed that they cannot enter existing product with same Invoice\_ID. If it doesn’t exist, then the record will be entered.

# View (Syntax)

Create view topdish(Prod\_info\_ID,number) as select Prod\_Info\_ID,sum(details\_qty) from Invoice\_Details group by prod\_info\_id

## **View (Screenshots)**

Within the view it shows how many times the product has been orderer



## **View (Explanation)**

This view is to display the top dish ordered by the customers.

# Sub queries (Syntax)

Create trigger inv\_totalPriceDel after delete on Invoice\_Details

referencing old as o

for each row mode db2sql

update Invoice

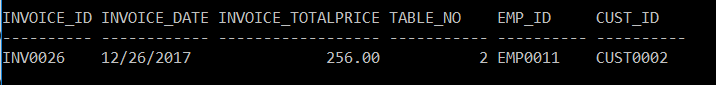
set Invoice\_TotalPrice = (select sum(Details\_Price) from Invoice\_Details

where Invoice\_Details.Invoice\_ID= o.Invoice\_ID)

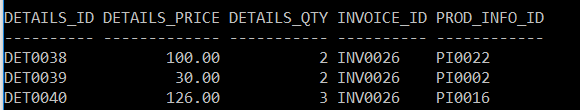
where Invoice.Invoice\_ID=o.Invoice\_ID

## **Sub queries (Screenshots)**

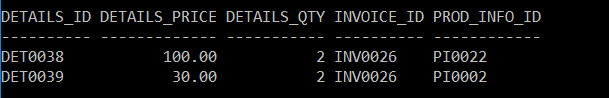
Invoice total price before delete in Invoice\_Details:



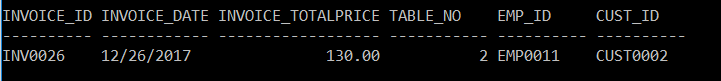
Invoice Details before delete:



Invoice Details after delete:



Invoice total price after delete in Invoice\_Details:



## **Sub queries (Explanation)**

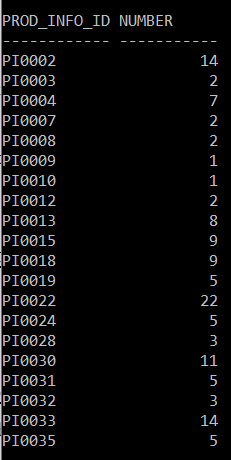
This is a trigger with sub queries implemented. This trigger will trigger after a record is deleted from Invoice\_Details table. If a record is deleted from Invoice\_Details, it will cause the total price in Invoice to change hence this trigger will recalculate the total price in Invoice.

# Query not covered 1 (Syntax)

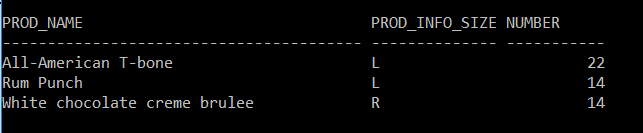
Select Prod\_name, prod\_info\_size, number from menu, product\_info, topdish where menu.prod\_ID = product\_info.prod\_ID and product\_info.prod\_info\_id = topdish.prod\_info\_ID order by number desc fetch first 3 rows only

## **Query not covered 1 (Screenshots)**

Values in the view ‘Topdish’



Using the select query above:



## **Query not covered 1 (Explanation)**

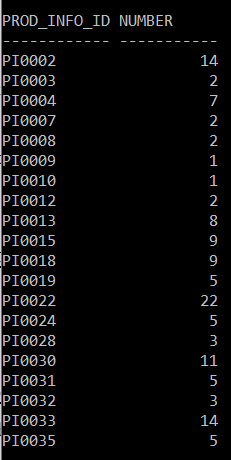
This query is paired with the query ‘fetch first 3 rows only’ to display the top 3 dish ordered from the view topdish.

# Query not covered 2 (Syntax)

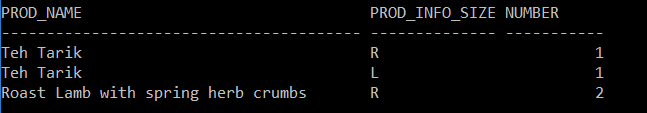
Select Prod\_name, prod\_info\_size, number from menu, product\_info, topdish where menu.prod\_ID = product\_info.prod\_ID and product\_info.prod\_info\_id = topdish.prod\_info\_ID order by number asc limit 3

## **Query not covered 2 (Screenshots)**

Values in the view ‘Topdish’:



Using the select query above:



## **Query not covered 2 (Explanation)**

This query is paired with the query ‘limit’ to display the lower 3 dish order from the view topdish.

# Value Insertion Query (Customer)

insert into Customer values(‘CUST0001’,’John Philips’,’0114724586’,’Selangor’,’Kuala Lumpur’, ’Jalan Mahmud’,’68100’)

insert into Customer values (‘CUST0002’,’Jacob Henderson’,’0178996734’,’Pulau Pinang’,’Georgetown’, ’Jalan Brother James’,’107450’)

insert into Customer values (‘CUST0003’,’Hong Zhe Yean’,’0183459870’,’Kedah’,’Sungai Petani’, ’Jalan Clover 6’,’433575’)

insert into Customer values(‘CUST0004’,’Lim Yiaw Hoong’,’0127778780’,’Selangor’,’Batu Caves’, ’Jalan Ehsan’,’25077’)

insert into Customer values(‘CUST0005’,’Muhammad Fitri bin Kamarudin’,’0134531298’, ’Kelantan’, ’Bachok District’, ’Jalan Pantai Aman’,’56080’)

insert into Customer values(‘CUST0006’,’Muhammad Firzani bin Fauzi’,’0165739855’, ‘Selangor’, ’Klang’, ’Jalan Goh Hock Huat’,’41400’)

insert into Customer values(‘CUST0007’,’Somraaj a/l Muthu’,’0182213455’,’Pulau Pinang’, ’Bayan Lepas’, ’Lebuh Sungai Ara 1’,’11900’)

insert into Customer values(‘CUST0008’,’Nancy Flowers’,’0124548886’,’Pulau Pinang’,’Gelugor’,’Jalan Permai’,’11700’)

insert into Customer values(‘CUST0009’,’Vignesh Varqa’,’0138907890’,’Selangor’,’Puchong’, ’Persiaran Kewajipan’,’47600’)

insert into Customer values(‘CUST0010’,’Muhammad Shazwan bin Rosli’,’0187734590’, ’Johor’, ’Senai’, ’Jalan Murni 12’,’81400’)

insert into Customer values(‘CUST0011’,’Soganraaj a/l Muniandy’,’0128890456’,’Johor’, ’Muar’, ’Jalan Sungai Abong’,’84000’)

insert into Customer values(‘CUST0012’,’Christopher Nolan’,’0164557890’, ’Pulau Pinang’, ’Georgetown’, ’Jalan Gurney Drive’,’10470’)

insert into Customer values(‘CUST0013’,’Leonard Ng’,’0139906566’,’Pulau Pinang’, ’Bayan Lepas’, ’Pintasan Kenari’,’11900’)

insert into Customer values(‘CUST0014’,’Daryl Cheah’,’0182215580’,’Selangor’,’Kapar’, ’Jalan Mohammad Ikram’,’45600’)

insert into Customer values(‘CUST0015’,’Muhammad Afiq bin Ali’,’0145368877’, ’Kelantan’, ’Kuala Krai’, ’Jalan Mutiara’,’18050’)

insert into Customer values(‘CUST0016’, ’Muhammad Aminah binti Samsudin’,’0128904561’,’Pahang’,’Kuantan’,’Jalan Sungai Lembing’,’28000’)

insert into Customer values(‘CUST0017’,’Yasmin Yacob’,’0168984572’,’Negeri Sembilan’, ’Gemas’, ’Jalan DS 3’,’73400’)

insert into Customer values(‘CUST0018’,’George Lucas’,’0136722385’,’Perlis’,’Kangar’, ’Persiaran Jubli Emas’,’01000’)

# Insert Query (Employee)

Insert into Employee values (‘EMP0001’,’Waylon Dalton’, ’Manager’)

Insert into Employee values (‘EMP0002’,’Teng Hock Lian’, ’Excutive Chef’)

Insert into Employee values (‘EMP0003’,’Mary Jane’, ’Sous Chef’)

Insert into Employee values (‘EMP0004’,’Muhammad Mohktar bin Khairi’,’Pastry Chef’)

Insert into Employee values (‘EMP0005’,’Somendiran a/l Muniandy’,’Soup and sauce Cook’)

Insert into Employee values (‘EMP0006’,’James Lee’, ’Sous Chef ’)

Insert into Employee values (‘EMP0007’,’Taro Misaki’,’Sous Chef’)

Insert into Employee values (‘EMP0008’,’Robin Dubos’, ’Fry Cook’)

Insert into Employee values (‘EMP0009’,’Hew Choon Yen’, ’Broiler Cook’)

Insert into Employee values (‘EMP0010’,’Dmitri Oleksander’,’Kitchen Manager’)

Insert into Employee values (‘EMP0011’,’Mohammad Taufik bin Umar’,’Waiter’)

Insert into Employee values (‘EMP0012’,’Hong Ming Kean’,’Waiter’)

Insert into Employee values (‘EMP0013’,’Lee Hui Yin’,’Waiter’)

Insert into Employee values (‘EMP0014’,’Frank Masson’,’Waiter’)

Insert into Employee values (‘EMP0015’,’Muhammad Khadija binti Omar’,’Waiter’)

# Insert Query (Cuisine\_Type)

insert into Cuisine\_Type values(‘CT0001’,’American’)

insert into Cuisine\_Type values(‘CT0002’,’British’)

insert into Cuisine\_Type values(‘CT0003’,’Carribean’)

insert into Cuisine\_Type values(‘CT0004’,’Chinese’)

insert into Cuisine\_Type values(‘CT0005’,’French’)

insert into Cuisine\_Type values(‘CT0006’,’Greek’)

insert into Cuisine\_Type values(‘CT0007’,’Indian’)

insert into Cuisine\_Type values(‘CT0008’,’Italian’)

insert into Cuisine\_Type values(‘CT0009’,’Japanese’)

insert into Cuisine\_Type values(‘CT0010’,’Malaysian’)

insert into Cuisine\_Type values(‘CT0011’,’Mediterranean’)

insert into Cuisine\_Type values(‘CT0012’,’Mexican’)

insert into Cuisine\_Type values(‘CT0013’,’Spanish’)

insert into Cuisine\_Type values(‘CT0014’,’Thai’)

insert into Cuisine\_Type values(‘CT0015’,’Turkish’)

insert into Cuisine\_Type values(‘CT0016’,’Vietnamese’ )

# Insert Query (Menu)

insert into Menu values(‘P0001’,’Rum Punch’,’Beverage’,’CT0003’)

insert into Menu values(‘P0002’,’Roast lamb with spring herbs crumbs’,’Food’,’CT0002’)

insert into Menu values(‘P0003’,’British pork cassoulet’,’Food’,’CT0002’)

insert into Menu values(‘P0004’,’Nasi Lemak’,’Food’,’CT0010’)

insert into Menu values(‘P0005’,’Teh Tarik’,’Beverage’,’CT0010’)

insert into Menu values(‘P0006’,’Ambra and Sour Plum Juice’,’Beverage’,’CT0010’)

insert into Menu values(‘P0007’,’Beef Rendang’,’Food’,’CT0010’)

insert into Menu values(‘P0008’,’Fish head curry’,’Food’,’CT0010’)

insert into Menu values(‘P0009’,’Key lime pie’,’Desert’,’CT0001’)

insert into Menu values(‘P0010’,’Sticky ribs with roast potato salad’,’Food’,’CT0001’)

insert into Menu values(‘P0011’,’All-American T-bone’,’Food’,’CT0001’)

insert into Menu values(‘P0012’,’Fish tacos with Baja sauce’,’Food’,’CT0012’)

insert into Menu values(‘P0013’,’Michelada’,’Beverage’,’CT0012’)

insert into Menu values(‘P0014’,’Miso Soup’,’Food’,’CT0009’)

insert into Menu values(‘P0015’,’Katsu pork with sticky rice’,’Food’,’CT0009’)

insert into Menu values(‘P0016’,’Green tea frozen yogurt with sesame’,’Desert’,’CT0009’)

insert into Menu values(‘P0017’,’White chocolate creme brulee’,’Desert’,’CT0005’)

insert into Menu values(‘P0018’,’Citron Presse’,’Beverage’,’CT0005’)

# Insert Query (Product\_Info)

Insert into product\_info values(‘PI0001’,’R’,’10.00’,’P0001’)

Insert into product\_info values(‘PI0002’,’L’,’15.00’,’P0001’)

Insert into product\_info values(‘PI0003’,’R’,’25.00’,’P0002’)

Insert into product\_info values(‘PI0004’,’L’,’32.00’,’P0002’)

Insert into product\_info values(‘PI0005’,’R’,’26.00’,’P0003’)

Insert into product\_info values(‘PI0006’,’L’,’32.00’,’P0003’)

Insert into product\_info values(‘PI0007’,’R’,’13.50’,’P0004’)

Insert into product\_info values(‘PI0008’,’L’,’16.00’,’P0004’)

Insert into product\_info values(‘PI0009’,’R’,’8.00’,’P0005’)

Insert into product\_info values(‘PI0010’,’L’,’10.00’,’P0005’)

Insert into product\_info values(‘PI0011’,’R’,’11.00’,’P0006’)

Insert into product\_info values(‘PI0012’,’L’,’13.50’,’P0006’)

Insert into product\_info values(‘PI0013’,’R’,’24.50’,’P0007’)

Insert into product\_info values(‘PI0014’,’L’,’31.50’,’P0007’)

Insert into product\_info values(‘PI0015’,’R’,’35.00’,’P0008’)

Insert into product\_info values(‘PI0016’,’L’,’42.00’,’P0008’)

Insert into product\_info values(‘PI0017’,’R’,’23.50’,’P0009’)

Insert into product\_info values(‘PI0018’,’L’,’30.00’,’P0009’)

Insert into product\_info values(‘PI0019’,’R’,’36.00’,’P0010’)

Insert into product\_info values(‘PI0020’,’L’,’42.00’,’P0010’)

Insert into product\_info values(‘PI0021’,’R’,’40.00’,’P0011’)

Insert into product\_info values(‘PI0022’,’L’,’50.00’,’P0011’)

Insert into product\_info values(‘PI0023’,’R’,’23.50’,’P0012’)

Insert into product\_info values(‘PI0024’,’L’,’30.00’,’P0012’)

Insert into product\_info values(‘PI0025’,’R’,’17.00’,’P0013’)

Insert into product\_info values(‘PI0026’,’L’,’20.00’,’P0013’)

Insert into product\_info values(‘PI0027’,’R’,’12.50’,’P0014’)

Insert into product\_info values(‘PI0028’,’L’,’14.50’,’P0014’)

Insert into product\_info values(‘PI0029’,’R’,’24.00’,’P0015’)

Insert into product\_info values(‘PI0030’,’L’,’28.00’,’P0015’)

Insert into product\_info values(‘PI0031’,’R’,’16.00’,’P0016’)

Insert into product\_info values(‘PI0032’,’L’,’20.00’,’P0016’)

Insert into product\_info values(‘PI0033’,’R’,’23.00’,’P0017’)

Insert into product\_info values(‘PI0034’,’L’,’26.50’,’P0017’)

Insert into product\_info values(‘PI0035’,’R’,’12.00’,’P0018’)

Insert into product\_info values(‘PI0036’,’L’,’15.50’,’P0018’)

# Insert query (Invoice)

Insert into invoice values (‘INV0001’,’2017-12-23’,0,2,’EMP0011’,’CUST0001’)

Insert into invoice values (‘INV0002’,’2017-12-23’,0,3,’EMP0012’,’CUST0002’)

Insert into invoice values (‘INV0003’,’2017-12-23’,0,1,’EMP0015’,’CUST0003’)

Insert into invoice values (‘INV0004’,’2017-12-23’,0,4,’EMP0011’,’CUST0004’)

Insert into invoice values (‘INV0005’,’2017-12-23’,0,6,’EMP0014’,’CUST0005’)

Insert into invoice values (‘INV0006’,’2017-12-23’,0,2,’EMP0013’,’CUST0006’)

Insert into invoice values (‘INV0007’,’2017-12-24’,0,1,’EMP0013’,’CUST0007’)

Insert into invoice values (‘INV0008’,’2017-12-24’,0,3,’EMP0014’,’CUST0002’)

Insert into invoice values (‘INV0009’,’2017-12-24’,0,2,’EMP0011’,’CUST0008’)

Insert into invoice values (‘INV0010’,’2017-12-24’,0,5,’EMP0014’,’CUST0009’)

Insert into invoice values (‘INV0011’,’2017-12-24’,0,6,’EMP0015’,’CUST0001’)

Insert into invoice values (‘INV0012’,’2017-12-24’,0,8,’EMP0013’,’CUST0010’)

Insert into invoice values (‘INV0013’,’2017-12-24’,0,7,’EMP0012’,’CUST0011’)

Insert into invoice values (‘INV0014’,’2017-12-24’,0,3,’EMP0011’,’CUST0012’)

Insert into invoice values (‘INV0015’,’2017-12-24’,0,1,’EMP0014’,’CUST0013’)

Insert into invoice values (‘INV0016’,’2017-12-26’,0,2,’EMP0011’,’CUST0004’)

Insert into invoice values (‘INV0017’,’2017-12-26’,0,1,’EMP0015’,’CUST0009’)

Insert into invoice values (‘INV0018’,’2017-12-26’,0,7,’EMP0014’,’CUST0014’)

Insert into invoice values (‘INV0019’,’2017-12-26’,0, 5,’EMP0011’,’CUST0015’)

Insert into invoice values (‘INV0020’,’2017-12-26’,0, 2,’EMP0012’,’CUST0016’)

Insert into invoice values (‘INV0021’,’2017-12-26’,0, 6,’EMP0013’,’CUST0017’)

Insert into invoice values (‘INV0022’,’2017-12-26’,0, 6,’EMP0015’,’CUST0001’)

Insert into invoice values (‘INV0023’,’2017-12-26’,0, 4,’EMP0014’,’CUST0009’)

Insert into invoice values (‘INV0024’,’2017-12-26’,0, 3,’EMP0012’,’CUST0010’)

Insert into invoice values (‘INV0025’,’2017-12-26’,0, 2,’EMP0013’,’CUST0018’)

# Insert Query (Invoice\_Details)

Call check\_details(‘DET0001’,0,2,’INV0001’,’PI0002’)

Call check\_details(‘DET0002’,0,1,’INV0001’,’PI0009’)

Call check\_details(‘DET0003’,0,3,’INV0002’,’PI0002’)

Call check\_details(‘DET0004’,0,1,’INV0002’,’PI0003’)

Call check\_details(‘DET0005’,0,2,’INV0003’,’PI0007’)

Call check\_details(‘DET0006’,0,4,’INV0004’,’PI0030’)

Call check\_details(‘DET0007’,0,2,’INV0005’,’PI0033’)

Call check\_details(‘DET0008’,0,1,’INV0005’,’PI0022’)

Call check\_details(‘DET0009’,0,2,’INV0006’,’PI0024’)

Call check\_details(‘DET0010’,0,3,’INV0007’,’PI0019’)

Call check\_details(‘DET0011’,0,5,’INV0008’,’PI0035’)

Call check\_details(‘DET0012’,0,5,’INV0008’,’PI0004’)

Call check\_details(‘DET0013’,0,2,’INV0009’,’PI0028’)

Call check\_details(‘DET0014’,0,2,’INV0010’,’PI0015’)

Call check\_details(‘DET0015’,0,3,’INV0011’,’PI0032’)

Call check\_details(‘DET0016’,0,1,’INV0011’,’PI0022’)

Call check\_details(‘DET0017’,0,3,’INV0012’,’PI0013’)

Call check\_details(‘DET0018’,0,6,’INV0013’,’PI0030’)

Call check\_details(‘DET0019’,0,2,’INV0014’,’PI0018’)

Call check\_details(‘DET0020’,0,2,’INV0015’,’PI0012’)

Call check\_details(‘DET0021’,0,2,’INV0015’,’PI0019’)

Call check\_details(‘DET0022’,0,1,’INV0016’,’PI0010’)

Call check\_details(‘DET0023’,0,1,’INV0016’,’PI0003’)

Call check\_details(‘DET0024’,0,2,’INV0017’,’PI0008’)

Call check\_details(‘DET0025’,0,3,’INV0018’,’PI0024’)

Call check\_details(‘DET0026’,0,5,’INV0019’,’PI00031’)

Call check\_details(‘DET0027’,0,5,’INV0019’,’PI0013’)

Call check\_details(‘DET0028’,0,1,’INV0020’,’PI0028’)

Call check\_details(‘DET0029’,0,1,’INV0020’,’PI0030’)

Call check\_details(‘DET0030’,0,7,’INV0021’,’PI0015’)

Call check\_details(‘DET0031’,0,2,’INV0022’,’PI0033’)

Call check\_details(‘DET0032’,0,7,’INV0023’,’PI0018’)

Call check\_details(‘DET0033’,0,2,’INV0023’,’PI0004’)

Call check\_details(‘DET0034’,0,8,’INV0024’,’PI0022’)

Call check\_details(‘DET0035’,0,7,’INV0024’,’PI0002’)

Call check\_details(‘DET0036’,0,10,’INV0025’,’PI0022’)

Call check\_details(‘DET0037’,0,10,’INV0025’,’PI0033’)