



Gino's Pizza

Restaurant Order Management System

Team Project Phase 3 Report

By

Archana Joshi

Karis Kim

Nivedita Talole

Sonal Thakre

Trang Vu

Table of Contents

1. Phase 1

1.1. Overview	3
1.2. Scope	3
1.3. Assumptions	3
1.4. Roles	4
1.5. Business Requirements	4-6
1.6. Definitions	6
1.7. Summary	6-7

2. Phase 2

2.1. Changes from Phase 1 Report	7
2.2. Entity Relationship Diagram	8
2.3. Relational Model	9

3. Phase 3

3.1. Table Creation Scripts	10-12
3.2. Inserting Data into All Tables	13-24
3.3. Queries	25-35
3.4. Workflow Demonstrations	36-38

Phase 1

1.1 Overview

Gino's Pizza is a fast-casual restaurant located in Marietta. The restaurant is known for its delicious pizza and friendly service. With the growth of the business, the need to create a database management system that helps the restaurant improve productivity and optimize daily operations became evident. The proposed database management system is designed to assist the restaurant's order management by providing automated functionalities such as sales, supplies, and purchase orders.

This document outlines the results of the basic business requirement analysis relevant to the order management database system, including the scope, assumptions, roles, business rules and a brief summary of the system.

1.2 Scope

This DBMS is designed mainly for restaurant order management only, and therefore, will not include rent, utilities, and financial functions outside the scope of order management. The exception will be employee information such as contact and salary information, even though they do not explicitly fall under the order management category.

1.3 Assumptions

- Employees are not provided health care.
- Only the manager can place purchase orders to vendors.
- Employees are regularly and consistently updating inventory (i.e.- products/supplies that are being used).
- Once an order is fulfilled, the employee assigned to an order will check off the order status as "fulfilled."
- Customer should be able to place an order successfully via one of the declared methods.
- Vendors (a.k.a. suppliers) will send an invoice of the purchase order to the restaurant upon shipment.

1.4 Roles

User	Main Responsibilities
Manager	<ol style="list-style-type: none">1. Manage all supply order placements, vendor payments, hire/schedule of all employees, and salary payments.2. Record billing information from the vendor when a new order shipment arrives.3. System administration.
Chef	<ol style="list-style-type: none">1. Manage tasks related to prep, cooking and fulfilling orders.2. Update inventory of perishable/non-perishable supplies.
Staff (Cashier, Server, Kitchen)	<ol style="list-style-type: none">1. Take customer's orders.2. Place orders per to customer's requests.3. Process customer's payments.4. Fulfill completed orders (dine-in, pick-up or delivery).
Customer	<ol style="list-style-type: none">1. Place orders in person, by phone or online.2. Make payment of order by cash or credit card.3. May pick up, dine in or have it delivered.4. May create/use online account for orders.

1.5 Business Requirements

Customer

- Customer can place an order by phone, online or in person.
- Customer must register/create an online account to place orders online.
- Customer specifies the order type- whether the order is for pick-up, dine-in, or delivery- at the time of order.
- Once customer places an order, an order ID is assigned to include the date, order details (i.e.- menu items), total price, payment method and payment status (paid/unpaid).
- Phone and online orders can be paid at the time of order or at delivery/pick up.
- Customer may pay by cash or credit card.
- Customer with a large order may place the order days prior. In such case, the “order date” shall reflect the date the order is placed, and “order details date” shall reflect the date the order is fulfilled.
- Customer info retained by the business shall be:
 - Last & first name, phone number, email, address, credit card number
- Customer info may be obtained by:
 - From credit card info upon payment
 - Customer registers online (to place order)
 - Restaurant staff enters data at the restaurant

Employee

- Employees must be able to access the system using valid credentials.
- All employees shall be able to view a list of orders and their custom pizzas.
- All employees must work 30-40 hours a week.
- Staff must be able to place and process an order for customer.
- Manager shall have employee's personal info, such as: last & first name, phone number, SSN, ID, email, address, birthdate, salary.
- Manager shall handle all salaries, bills, vendor payments, supply orders, hiring and schedules of all staff.
- Manager shall record the billing information from the vendor when a new order shipment arrives.
- All employees are full-time, but employees can either be staff or manager, not both simultaneously.

Order and Order Details

- Orders may be placed in person, by phone or online.
- Orders will be entered by the staff according to the customers.
- Order number will be assigned when a staff fills out an order entry.
- Each order will be assigned an employee who is responsible for ensuring that the order is fulfilled.
- There are 3 types of orders: dine-in, pick-up, or delivery.
- Order table, order date, menu item number (food items - this number should reference the item number from the menu), and quantity will also be recorded.
- The order system will track payment options (by cash or credit card only) and record every payment transaction.

Items

- Each item shall be assigned an item number. The item number will be referenced in the order details when placing an order from a customer.
- Items will have the following attributes:
 - Item number (Pizza, Salad, Sides, Beverages)
 - Item name
 - Item descriptions
 - Price

Vendor

- The manager shall record the billing information from the vendor when a new order shipment comes in.
- For every shipment, manager shall enter the billing number provided by the vendor, vendor ID, supply ID, delivery date, product descriptions, and cost.

Supply

- All employees shall regularly track and update ingredients and supplies according to the threshold levels.
- Each supply shall have a unique ID, supply name, and quantity.
- Supply inventory may be organized in various ways – alphabetically, by cost or by category.

Purchase Order

- Only manager can place purchase orders.
- Manager can create multiple purchase orders to vendor.
- One purchase order shall be created by and associated with one employee, the manager.
- Vendor can have multiple purchase orders.
- One purchase order can have only one vendor.

1.6 Definitions

- *Customer* is defined as any person or organization that places an order for any item(s) sold by the restaurant and makes payment to purchase it.
- *Employee* is defined as any person hired by the manager to work for the restaurant and receive monetary compensation in return for the labor. Any non-paid volunteer workforce shall not be considered an employee.
- *Order* is defined as a customer-initiated request of menu items to purchase.
- *Item* is defined as a listing of food items available for sale at this business.
- *Vendor* (a.k.a. *supplier*) is defined as a person or company that sells perishable and non-perishable goods to enable the operation of this business.
- *Purchase Order* is defined as a restaurant-generated document that authorizes a purchase transaction including descriptions, quantities, prices, payment terms, date of shipment, and identifies a specific seller (a.k.a. vendor or supplier).
- *Supply* is defined as all ingredients necessary for creating items on the menu (e.g. - flour, butter, tomato, cheese, lettuce, coke), as well as all supplies necessary for fulfillment of the order (e.g. - napkins, utensils, to-go boxes, cups, plates, kitchen tools).

1.7 Summary

The goal of this restaurant order management system is to improve efficiency and productivity by automating the functions pertaining to order management. The proposed order management system will store, track, maintain, and secure records of customers, orders, employees, supplies, vendors and purchase orders. The system will provide a reliable, secure, and efficient method for record keeping, which will enable the restaurant to focus its resources on activities of greater importance than record keeping. The information from this system will also improve the

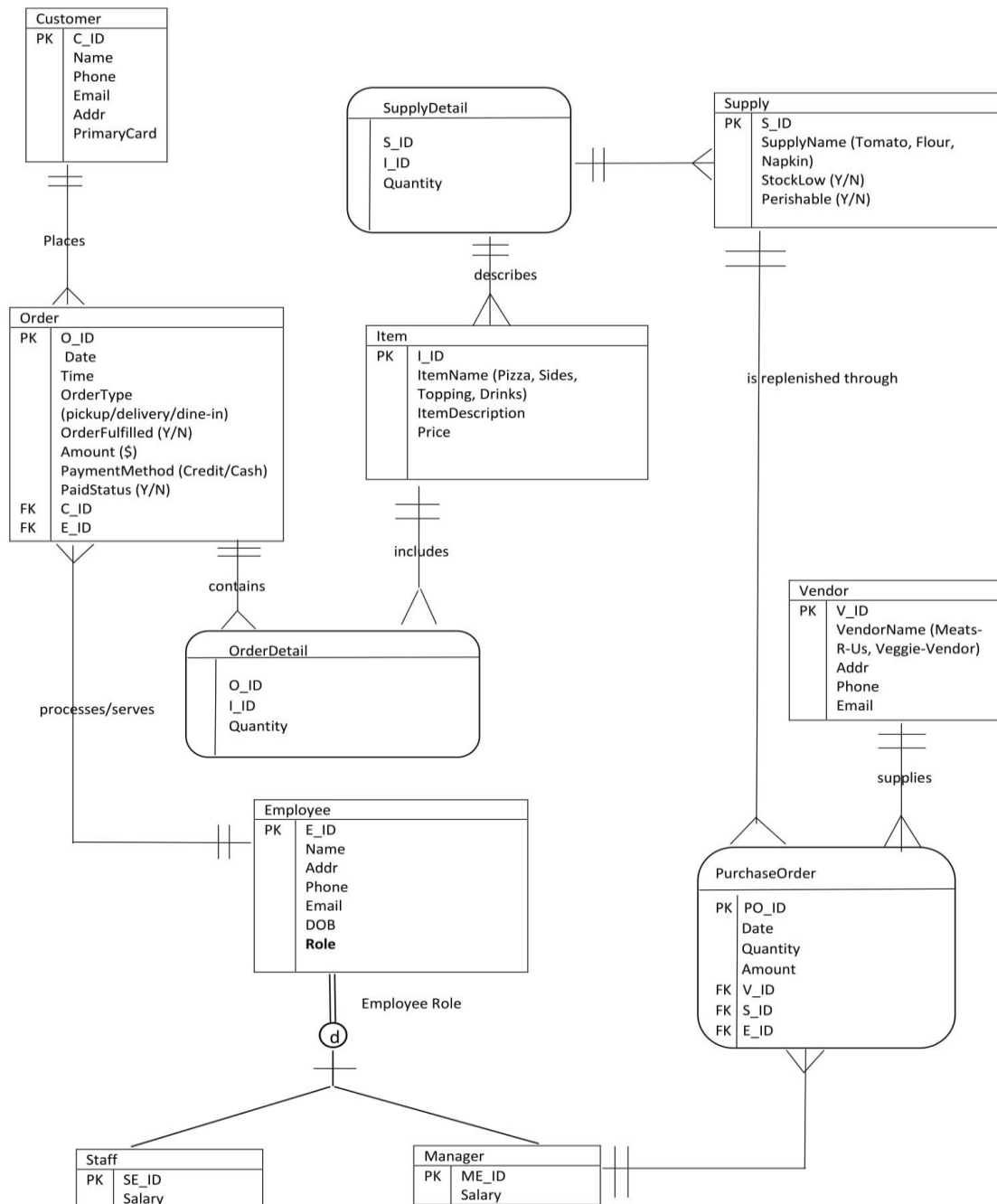
restaurant's ability with planning and management, through various robust query and reporting functions.

Phase 2

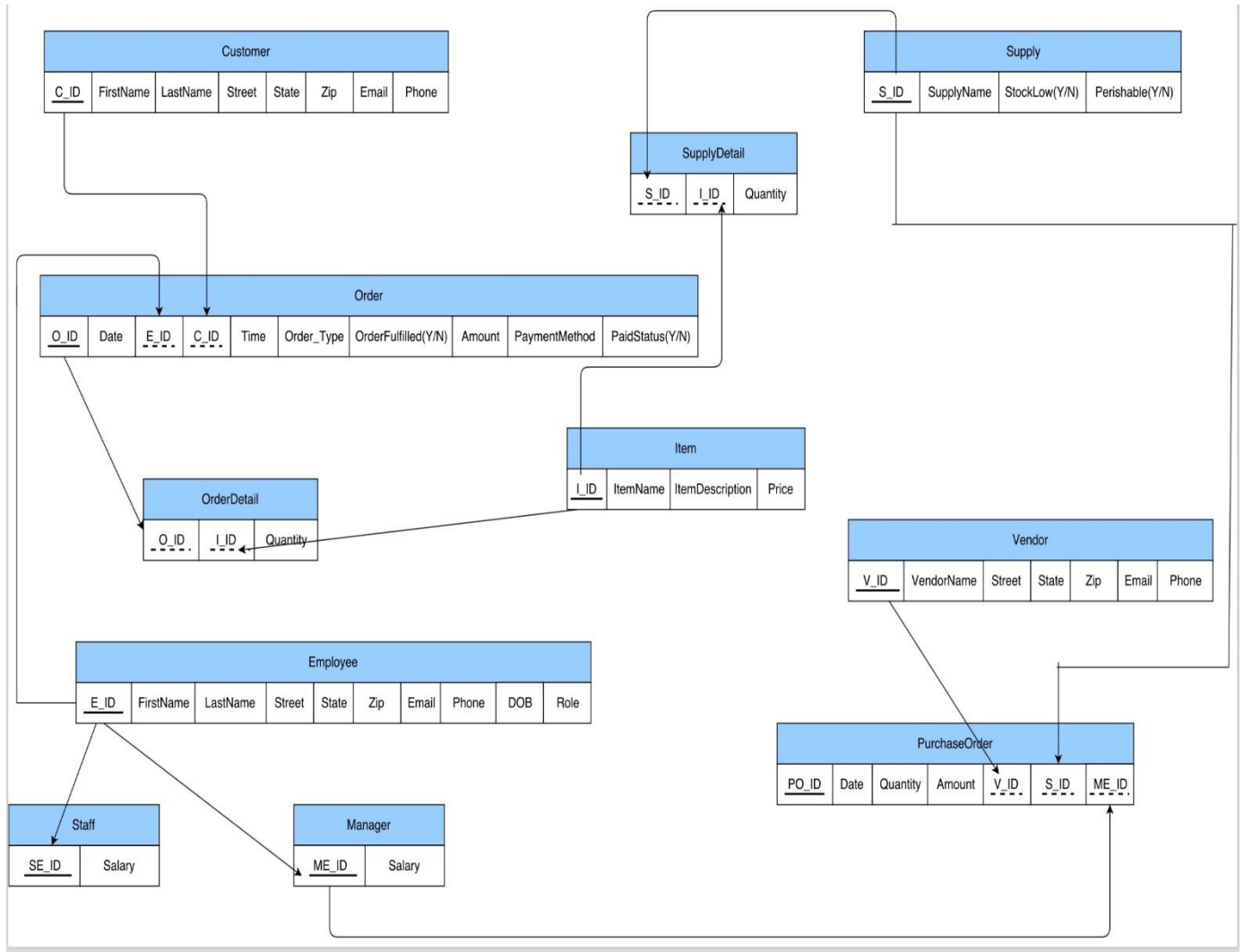
2.1 Changes from Phase 1 Report

- Only manager can place purchase orders.
- All employees are full-time, but employees can either be staff or manager, not both simultaneously
- "Menu" entity shall be referred to as "Item" in the DB.
- "Supplier" shall be referred to as "Vendor" in the DB.
- "Product" shall be referred to as "Supply" in the DB.

2.2 Entity Relationship Diagram



2.3 Relational Model



3. Phase 3

3.1. SQL Table Creation Scripts

Customer Table

```
Create Table Customer (  
Customer_ID varchar2 (10) not null,  
First_Name varchar2 (25) not null,  
Last_Name varchar2 (25) not null,  
Street varchar2 (50),  
City varchar2 (20),  
State varchar2 (20),  
Zip number (10,0),  
Email_ID varchar2 (30) not null,  
Phone number (15,0),  
Constraint Customer_ID Primary key (Customer_ID));
```

Order Table

```
Create table Orders (  
Order_ID varchar2(10),  
Order_Date date,  
Employee_ID varchar2(10),  
Customer_ID varchar2(10),  
Order_Time varchar2(10),  
Order_Type varchar2(20),  
Order_Fulfilled varchar2(5),  
Amount number (6,0),  
Payment_Method varchar2(15),  
Paid_Status varchar2(5),  
Constraint Order_ID Primary Key(Order_ID),  
Constraint Employee_ID_FK Foreign Key (Employee_ID) references Employee  
(Employee_ID),  
Constraint Customer_ID_FK Foreign Key(Customer_ID) references Customer(Customer_ID));
```

Order Detail Table

```
Create table OrderDetail (  
Order_ID varchar2(10),  
Item_ID varchar2(10),  
Quantity number (3,0),  
constraint OrderDetail_PK Primary Key (Order_ID, Item_ID),  
Constraint OrderDetail_Order Foreign Key(Order_ID) references Orders(Order_ID),  
Constraint OrderDetail_Item foreign key(Item_ID) references Item(Item_ID));
```

Employee Table

```
Create table Employee (  
Employee_ID varchar2 (10) not null,  
First_Name varchar2 (25) not null,  
Last_Name varchar2 (25) not null,  
Street varchar2 (50),  
City varchar2 (20),  
State varchar2 (20),  
Zip number (10, 0),  
Email_ID varchar2 (50),  
Phone number (15, 0),  
DOB varchar2 (15),  
Role varchar2 (10),  
Constraint Employee_ID Primary Key (Employee_ID));
```

Staff Table

```
Create table Staff (  
Staff_ID varchar2 (10),  
Salary number (10,0),  
Constraint Staff_ID Primary Key (Staff_ID),  
Constraint Staff_IDFK foreign key (Staff_ID) references Employee (Employee_ID));
```

Manager Table

```
Create table Manager (  
Manager_ID varchar2(10),  
Salary number (10,0),  
Constraint Manager_ID Primary Key (Manager_ID),  
Constraint Manager_ID_FK foreign key (Manager_ID) references Employee (Employee_ID));
```

Supply Detail Table

```
Create table SupplyDetail (  
Supply_ID varchar2(10),  
Item_ID varchar2(10),  
Quantity number (3,0),  
constraint SupplyDetail_PK Primary Key (Supply_ID, Item_ID),  
Constraint SupplyDetail_Supply Foreign Key (Supply_ID) references Supply (Supply_ID),  
Constraint SupplyDetail_Item foreign key (Item_ID) references Item (Item_ID));
```

Item Table

```
Create table Item (  
Item_ID varchar2(10),  
Item_Name varchar2(25),  
Item_Description varchar2(100),  
Price varchar2(5),  
Constraint Item_ID Primary Key(Item_ID));
```

Supply Table

```
Create table Supply (  
Supply_ID varchar2(10),  
Supply_Name varchar2(25),  
StockLow varchar2(5),  
Perishable varchar2(5),  
Constraint Supply_IDPK Primary Key(Supply_ID));
```

Vendor Table

```
Create table Vendor (  
Vendor_ID varchar2(10),  
Vendor_Name varchar2(25),  
Street varchar2(50),  
City varchar2(20),  
State varchar2(20),  
zip number (10,0),  
Email_ID varchar2(50),  
Phone number (15,0),  
Constraint Vendor_ID primary key(Vendor_ID));
```

Purchase Order Table

```
Create Table PurchaseOrder (  
PurchaseOrder_ID varchar2(10),  
PODate date,  
Quantity number (5,0),  
Amount number (5,0),  
Vendor_ID varchar2(10),  
Supply_ID varchar2(10),  
Manager_ID varchar2(10),  
Constraint PurchaseOrder_ID Primary Key(PurchaseOrder_ID),  
Constraint POVendor_ID Foreign Key (Vendor_ID) references Vendor (Vendor_ID),  
Constraint POSupply_ID Foreign Key (Supply_ID) references Supply (Supply_ID),  
Constraint POManager_ID Foreign Key(Manager_ID) references Manager(Manager_ID));
```

3.2 Inserting Data into All Tables

Customer Table

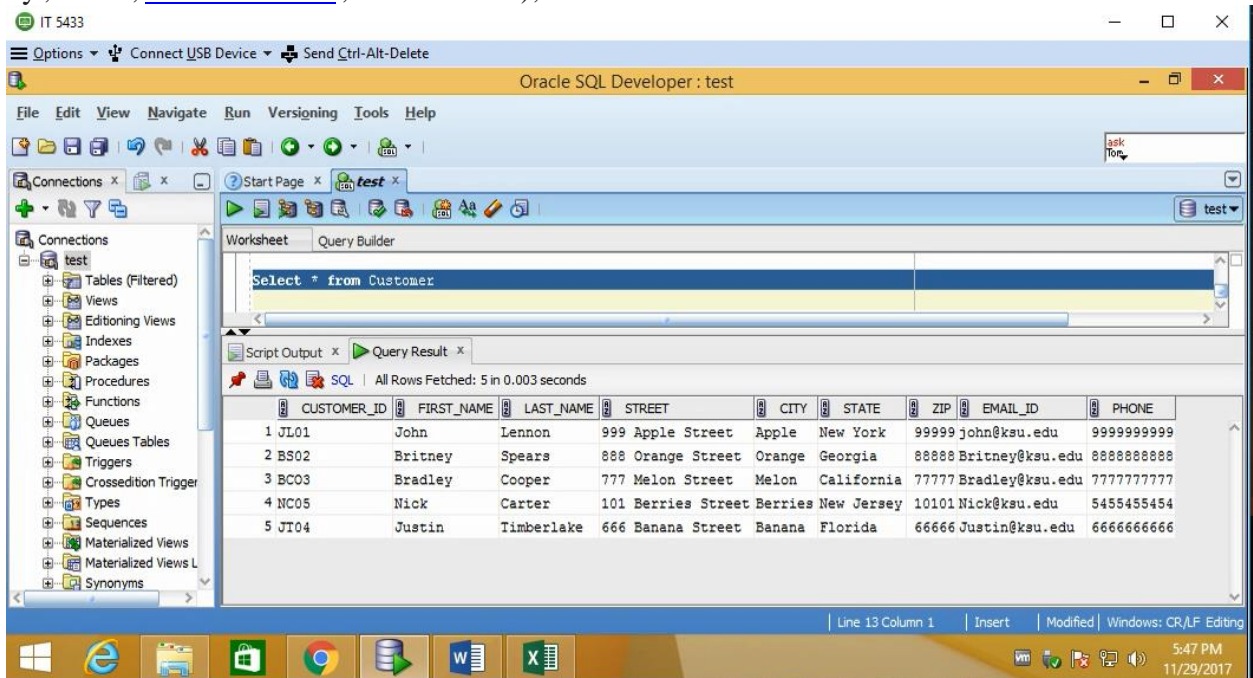
Insert into Customer (Customer_ID, First_Name, Last_Name, Street, City, State, Zip, Email_ID, Phone) values ('JL01', 'John','Lennon','999 Apple Street','Apple','New York','99999','John@ksu.edu','9999999999');

Insert into Customer (Customer_ID, First_Name, Last_Name, Street, City, State, Zip, Email_ID, Phone) values ('BS02','Britney','Spears','888 Orange Street','Orange','Georgia',888888,'Britney@ksu.edu','8888888888');

Insert into Customer (Customer_ID, First_Name, Last_Name, Street, City, State, Zip, Email_ID, Phone) values ('BC03','Bradley','Cooper','777 Melon Street','Melon','California',77777,'Bradley@ksu.edu','7777777777');

Insert into Customer (Customer_ID, First_Name, Last_Name, Street, City, State, Zip, Email_ID, Phone) values ('JT04','Justin','Timberlake','666 Banana Street','Banana','Florida',66666,'Justin@ksu.edu','6666666666');

Insert into Customer (Customer_ID, First_Name, Last_Name, Street, City, State, Zip, Email_ID, Phone) values ('NC05','Nick','Carter','101 Berries Street','Berries','New Jersey',10101,'Nick@ksu.edu','5455455454');



The screenshot shows the Oracle SQL Developer interface. The 'Connections' pane on the left lists a connection named 'test'. The 'Worksheet' pane displays the SQL query 'Select * from Customer'. The 'Query Result' pane shows the following data:

	CUSTOMER_ID	FIRST_NAME	LAST_NAME	STREET	CITY	STATE	ZIP	EMAIL_ID	PHONE
1	JL01	John	Lennon	999 Apple Street	Apple	New York	99999	john@ksu.edu	9999999999
2	BS02	Britney	Spears	888 Orange Street	Orange	Georgia	88888	Britney@ksu.edu	8888888888
3	BC03	Bradley	Cooper	777 Melon Street	Melon	California	77777	Bradley@ksu.edu	7777777777
4	NC05	Nick	Carter	101 Berries Street	Berries	New Jersey	10101	Nick@ksu.edu	5455455454
5	JT04	Justin	Timberlake	666 Banana Street	Banana	Florida	66666	Justin@ksu.edu	6666666666

Order Table

Insert into Orders (Order_ID, Order_Date, Employee_ID, Customer_ID, Order_Time, Order_Type, Order_Fulfilled, Amount, Payment_Method, Paid_Status) values ('O_105', (to_date('27/11/2017','DD/MM/YYYY')), 'E_JS05', 'NC05','10:15','Pick-Up','Yes',45,'Credit Card','Yes');

Insert into Orders (Order_ID, Order_Date, Employee_ID, Customer_ID, Order_Time, Order_Type, Order_Fulfilled, Amount, Payment_Method, Paid_Status) values ('O_104', (to_date('25/11/2017','DD/MM/YYYY')), 'E_RS04', 'JT04','14:45','Pick-Up','Yes',36,'Cash','Yes');

Insert into Orders (Order_ID, Order_Date, Employee_ID, Customer_ID, Order_Time, Order_Type, Order_Fulfilled, Amount, Payment_Method, paid_Status) values ('O_103', (to_date('21/11/2017','DD/MM/YYYY')), 'E_HD03', 'BCO3','12:30','Dine- In','Yes',112,'Credit Card','Yes');

Insert into Orders (Order_ID, Order_Date, Employee_ID, Customer_ID, Order_Time, Order_Type, Order_Fulfilled, Amount, Payment_Method, Paid_Status) values ('O_102', (to_date('18/11/2017','DD/MM/YYYY')), 'E_HM02', 'BS02','16:00','Take-Out','Yes',120,'Credit Card','Yes');

Insert into Orders (Order_ID, Order_Date, Employee_ID, Customer_ID, Order_Time, Order_Type, Order_Fulfilled, Amount, Payment_Method, Paid_Status) values ('O_101', (to_date('17/11/2017','DD/MM/YYYY')), 'E_RM01', 'JL01','11:00','Dine-in','Yes',54,'Cash','Yes');

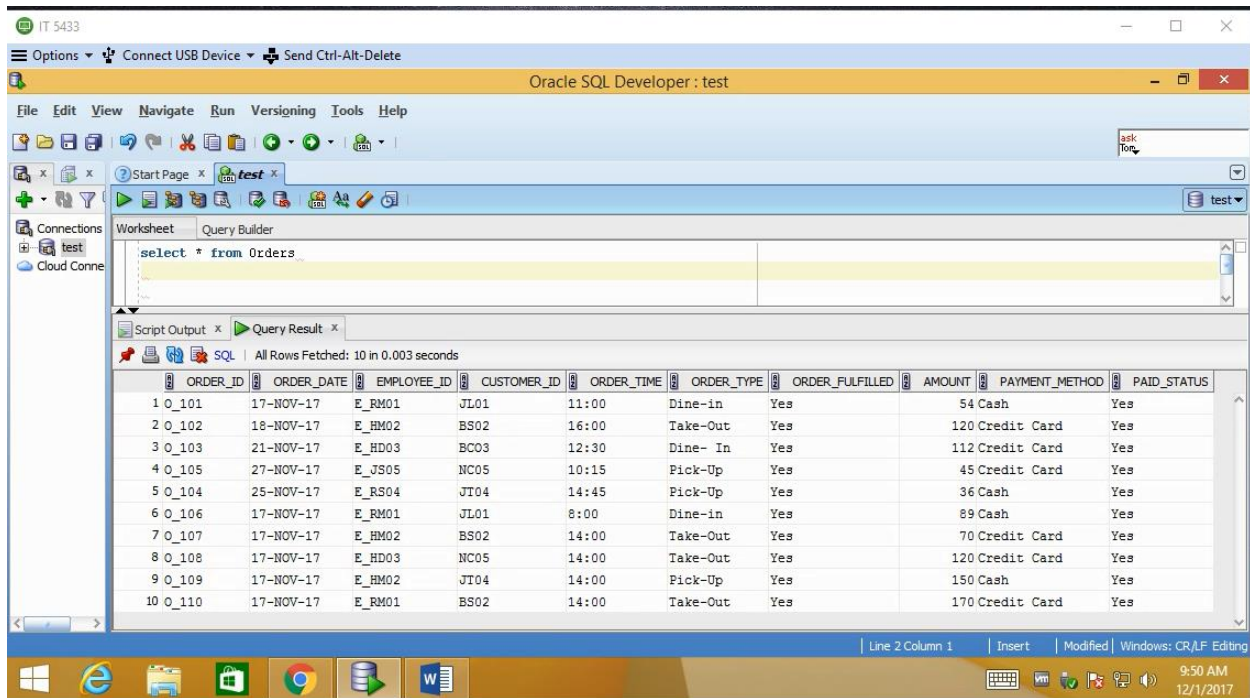
Insert into Orders (Order_ID, Order_Date, Employee_ID, Customer_ID, Order_Time, Order_Type, Order_Fulfilled, Amount, Payment_Method, Paid_Status) values ('O_106', (to_date('17/11/2017','DD/MM/YYYY')), 'E_RM01', 'JL01','8:00','Dine-in','Yes',89,'Cash','Yes');

Insert into Orders (Order_ID, Order_Date, Employee_ID, Customer_ID, Order_Time, Order_Type, Order_Fulfilled, Amount, Payment_Method, Paid_Status) values ('O_109', (to_date('17/11/2017','DD/MM/YYYY')), 'E_HM02', 'JT04','14:00','Pick-Up','Yes',150,'Cash','Yes');

Insert into Orders (Order_ID, Order_Date, Employee_ID, Customer_ID, Order_Time, Order_Type, Order_Fulfilled, Amount, Payment_Method, Paid_Status) values ('O_110', (to_date('17/11/2017','DD/MM/YYYY')), 'E_RM01', 'BS02','14:00','Take-Out','Yes',170,'Credit Card','Yes');

Insert into Orders (Order_ID, Order_Date, Employee_ID, Customer_ID, Order_Time, Order_Type, Order_Fulfilled, Amount, Payment_Method, Paid_Status) values ('O_108', (to_date('17/11/2017','DD/MM/YYYY')), 'E_HD03', 'NC05','14:00','Take-Out','Yes',120,'Credit Card','Yes');

Insert into Orders (Order_ID, Order_Date, Employee_ID, Customer_ID, Order_Time, Order_Type, Order_Fulfilled, Amount, Payment_Method, Paid_Status) values ('O_107', (to_date('17/11/2017','DD/MM/YYYY')), 'E_HM02', 'BS02','14:00','Take-Out','Yes',70,'Credit Card','Yes');



The screenshot shows the Oracle SQL Developer interface. The main window displays a query result for the 'Orders' table. The query is 'select * from Orders'. The result shows 10 rows of data. The columns are: ORDER_ID, ORDER_DATE, EMPLOYEE_ID, CUSTOMER_ID, ORDER_TIME, ORDER_TYPE, ORDER_FULFILLED, AMOUNT, PAYMENT_METHOD, and PAID_STATUS.

ORDER_ID	ORDER_DATE	EMPLOYEE_ID	CUSTOMER_ID	ORDER_TIME	ORDER_TYPE	ORDER_FULFILLED	AMOUNT	PAYMENT_METHOD	PAID_STATUS
1 O_101	17-NOV-17	E_RM01	JL01	11:00	Dine-in	Yes	54	Cash	Yes
2 O_102	18-NOV-17	E_HM02	BS02	16:00	Take-Out	Yes	120	Credit Card	Yes
3 O_103	21-NOV-17	E_HD03	BC03	12:30	Dine- In	Yes	112	Credit Card	Yes
4 O_105	27-NOV-17	E_JS05	NC05	10:15	Pick-Up	Yes	45	Credit Card	Yes
5 O_104	25-NOV-17	E_RS04	JT04	14:45	Pick-Up	Yes	36	Cash	Yes
6 O_106	17-NOV-17	E_RM01	JL01	8:00	Dine-in	Yes	89	Cash	Yes
7 O_107	17-NOV-17	E_HM02	BS02	14:00	Take-Out	Yes	70	Credit Card	Yes
8 O_108	17-NOV-17	E_HD03	NC05	14:00	Take-Out	Yes	120	Credit Card	Yes
9 O_109	17-NOV-17	E_HM02	JT04	14:00	Pick-Up	Yes	150	Cash	Yes
10 O_110	17-NOV-17	E_RM01	BS02	14:00	Take-Out	Yes	170	Credit Card	Yes

Order Detail Table

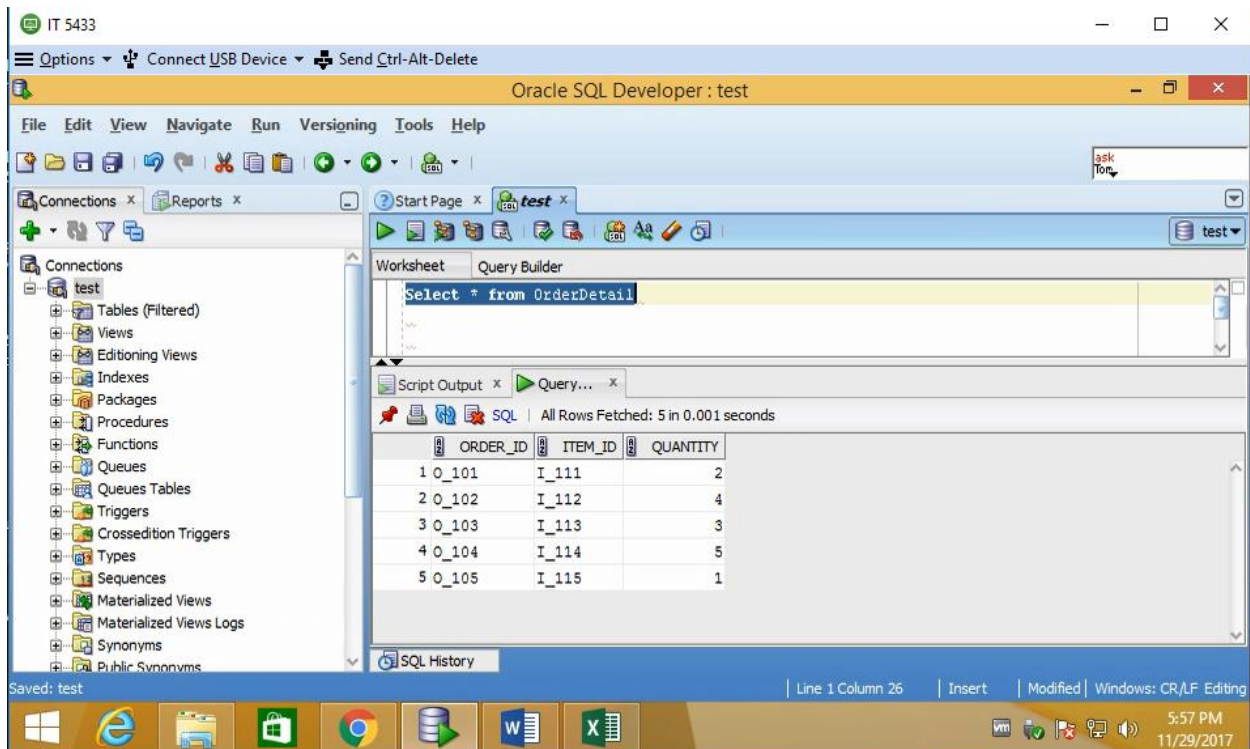
Insert into OrderDetail (Order_ID, Item_ID, Quantity) values ('O_101','I_111',2);

Insert into OrderDetail (Order_ID, Item_ID, Quantity) values ('O_102','I_112',4);

Insert into OrderDetail (Order_ID, Item_ID, Quantity) values ('O_103','I_113',3);

Insert into OrderDetail (Order_ID, Item_ID, Quantity) values ('O_104','I_114',5);

Insert into OrderDetail (Order_ID, Item_ID, Quantity) values ('O_105', 'I_115',1);



Employee Table

Insert into Employee(Employee_ID, First_Name, Last_Name, Street, City, State, Zip, Email_ID, Phone, DOB, Role)values('E_JS05','Jessica','Simpsons','111 Peach St', 'Peachtree', 'Georgia', 11111, 'Jessica1@ksu.edu','1111111111','5/12/1975','Manager');

Insert into Employee(Employee_ID, First_Name, Last_Name, Street, City, State, Zip, Email_ID, Phone, DOB, Role)values('E_RS04','Ren','Stevens','222 Hot St','Holler','Georgia',22222, 'Ren@ksu.edu', '2222222222','5/31/1993','Waiter');

Insert into Employee(Employee_ID, First_Name, Last_Name, Street, City, State, Zip, Email_ID, Phone, DOB, Role)values('E_HD03', 'Hilary', 'Duff','333 Middle st', 'Middleton', 'Georgia', 33333, 'Hilary@ksu.edu', '3333333333','4/12/1987','BarTender');

Insert into Employee(Employee_ID, First_Name, Last_Name, Street, City, State, Zip, Email_ID, Phone, DOB, Role)values('E_HM02', 'Hanna', 'Montana', '444 Front St', 'Dakota', 'Georgia', 44443, 'Hanna@ksu.edu', '4444444444', '6/5/1992','Hostess');

Insert into Employee(Employee_ID, First_Name, Last_Name, Street, City, State, Zip, Email_ID, Phone, DOB, Role)values('E_RM01', 'Ricky', 'Martin', '555 Back St', 'Atlanta', 'Georgia', 55555, 'Ricky@ksu.edu', '5555555555', '7/1/1960','Waiter');

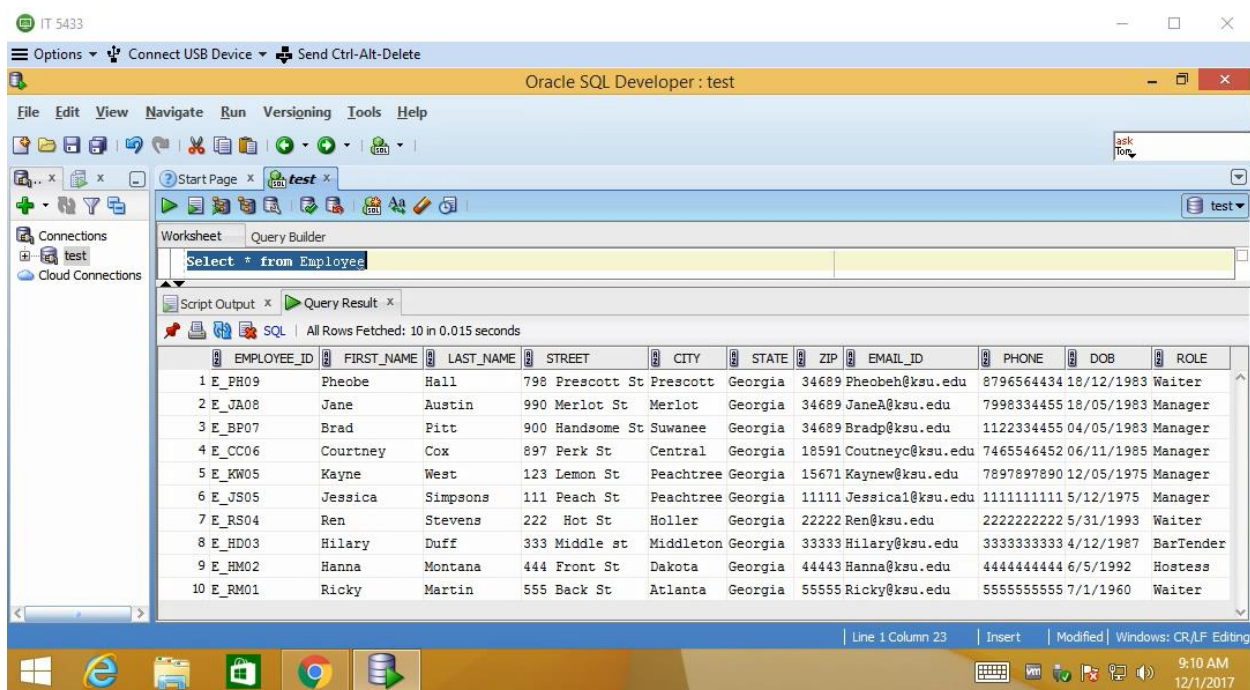

```
Insert into Employee(Employee_ID, First_Name, Last_Name, Street, City, State, Zip, Email_ID,
Phone, DOB, Role)values('E_PH09','Pheobe','Hall','798 Prescott St', 'Prescott', 'Georgia', 34689,
'Pheobeh@ksu.edu', '8796564434','18/12/1983','Waiter');
```

```
Insert into Employee(Employee_ID, First_Name, Last_Name, Street, City, State, Zip, Email_ID,
Phone, DOB, Role)values('E_JA08','Jane','Austin','990 Merlot St','Merlot','Georgia',
34689,'JaneA@ksu.edu', '7998334455','18/05/1983','Manager');
```

```
Insert into Employee(Employee_ID, First_Name, Last_Name, Street, City, State, Zip, Email_ID,
Phone, DOB, Role)values('E_BP07','Brad','Pitt','900 Handsome
St','Suwanee','Georgia',34689,'Bradp@ksu.edu', '1122334455','04/05/1983','Manager');
```

```
Insert into Employee(Employee_ID, First_Name, Last_Name, Street, City, State, Zip, Email_ID,
Phone, DOB, Role)values('E_CC06','Courtney','Cox','897 Perk
St','Central','Georgia',18591,'Coutneyc@ksu.edu', '7465546452','06/11/1985','Manager');
```

```
Insert into Employee(Employee_ID, First_Name, Last_Name, Street, City, State, Zip, Email_ID,
Phone, DOB, Role)values('E_KW05','Kayne','West','123 Lemon
St','Peachtree','Georgia',15671,'Kaynew@ksu.edu', '7897897890','12/05/1975','Manager')
```



IT 5433

Options ▾ Connect USB Device ▾ Send Ctrl-Alt-Delete

Oracle SQL Developer : test

File Edit View Navigate Run Versioning Tools Help

Start Page x test x

Connections test Cloud Connections

Worksheet Query Builder

Select * from Employee

Script Output x Query Result x

SQL All Rows Fetched: 10 in 0.015 seconds

EMPLOYEE_ID	FIRST_NAME	LAST_NAME	STREET	CITY	STATE	ZIP	EMAIL_ID	PHONE	DOB	ROLE
1 E_PH09	Pheobe	Hall	798 Prescott St	Prescott	Georgia	34689	Pheobeh@ksu.edu	8796564434	18/12/1983	Waiter
2 E_JA08	Jane	Austin	990 Merlot St	Merlot	Georgia	34689	JaneA@ksu.edu	7998334455	18/05/1983	Manager
3 E_BP07	Brad	Pitt	900 Handsome St	Suwanee	Georgia	34689	Bradp@ksu.edu	1122334455	04/05/1983	Manager
4 E_CC06	Courtney	Cox	897 Perk St	Central	Georgia	18591	Coutneyc@ksu.edu	7465546452	06/11/1985	Manager
5 E_KW05	Kayne	West	123 Lemon St	Peachtree	Georgia	15671	Kaynew@ksu.edu	7897897890	12/05/1975	Manager
6 E_JS05	Jessica	Simpsons	111 Peach St	Peachtree	Georgia	11111	Jessical@ksu.edu	1111111111	5/12/1975	Manager
7 E_RS04	Ren	Stevens	222 Hot St	Holler	Georgia	22222	Ren@ksu.edu	2222222222	5/31/1993	Waiter
8 E_HD03	Hilary	Duff	333 Middle st	Middleton	Georgia	33333	Hilary@ksu.edu	3333333333	4/12/1987	BarTender
9 E_HM02	Hanna	Montana	444 Front St	Dakota	Georgia	44443	Hanna@ksu.edu	4444444444	6/5/1992	Hostess
10 E_RM01	Ricky	Martin	555 Back St	Atlanta	Georgia	55555	Ricky@ksu.edu	5555555555	7/1/1960	Waiter

Line 1 Column 23 Insert Modified Windows: CR/LF Editing

9:10 AM 12/1/2017

Staff Table

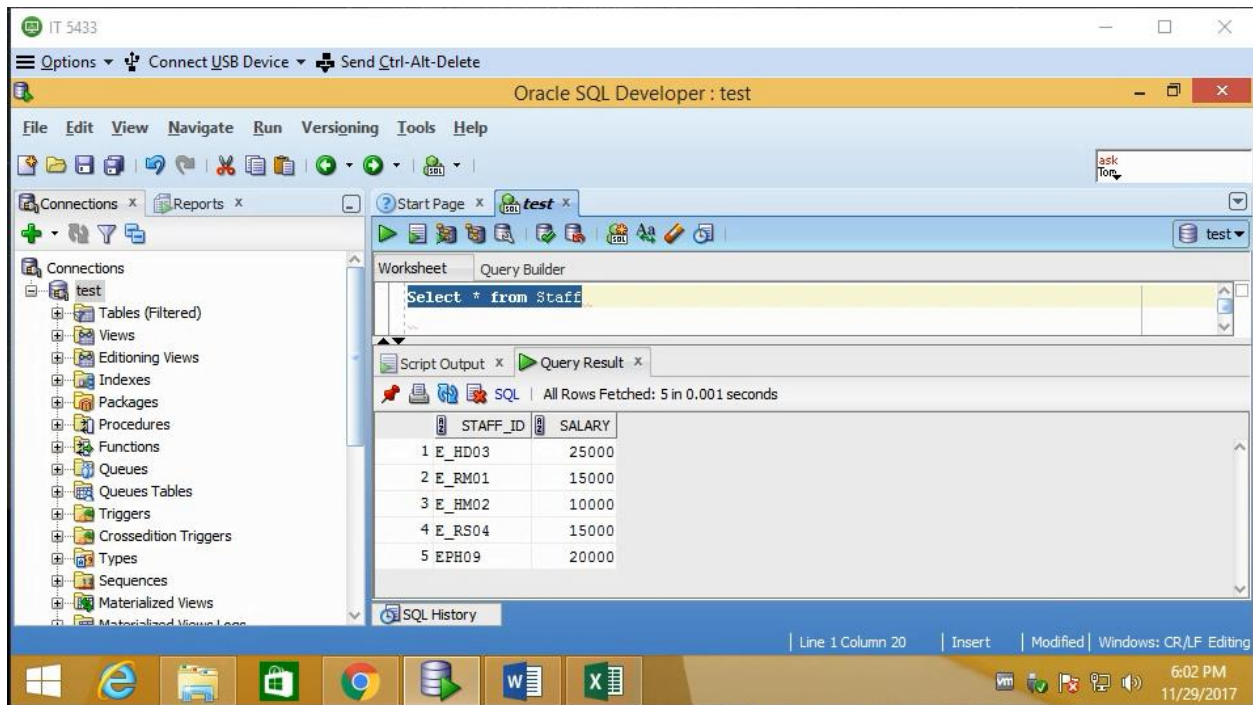
Insert into Staff (Staff_ID, Salary) values ('E_HD03', 25000);

Insert into Staff (Staff_ID, Salary) values ('E_RM01', 15000);

Insert into Staff (Staff_ID, Salary) values ('E_HM02', 10000);

Insert into Staff (Staff_ID, Salary) values ('E_RS04', 15000);

Insert into Staff (Staff_ID, Salary) values ('E_PH09', 20000);



Manager Table

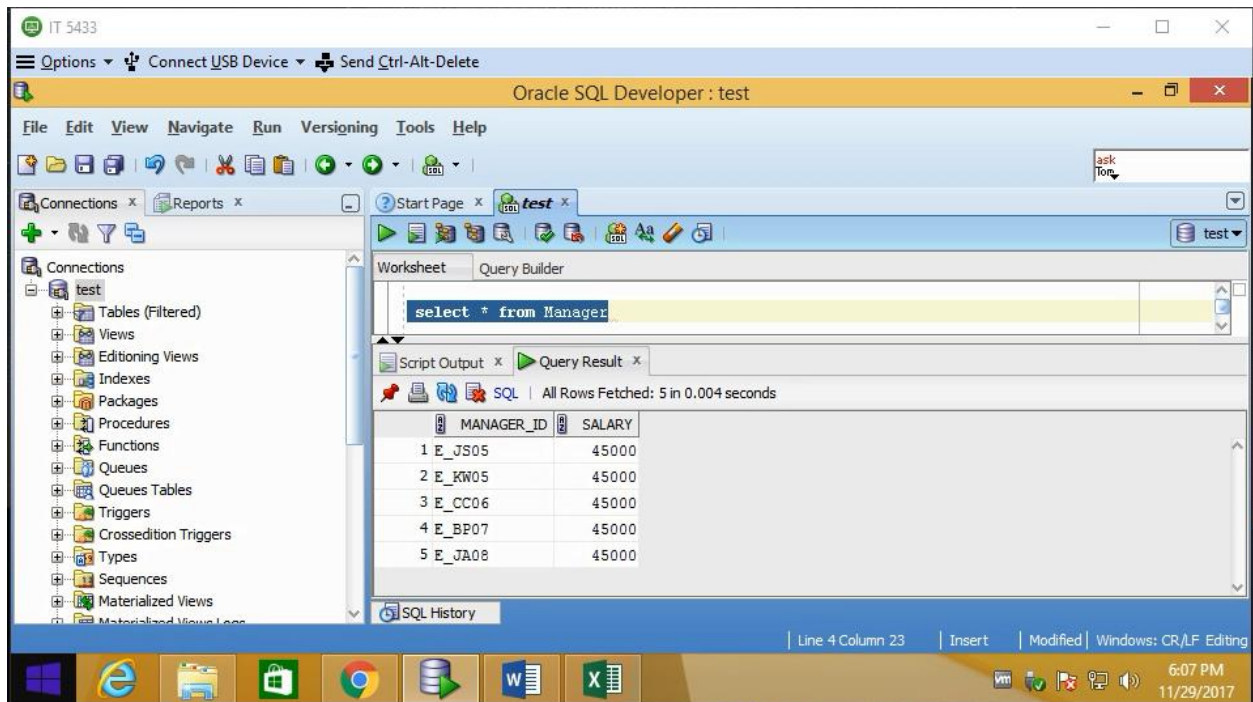
Insert into Manager (Manager_ID, Salary) values ('E_JS05', 40000);

Insert into Manager (Manager_ID, Salary) values ('E_KW05', 45000);

Insert into Manager (Manager_ID, Salary) values ('E_CC06', 50000);

Insert into Manager (Manager_ID, Salary) values ('E_BP07', 55000);

Insert into Manager (Manager_ID, Salary) values ('E_JA08', 60000);



Supply Detail Table

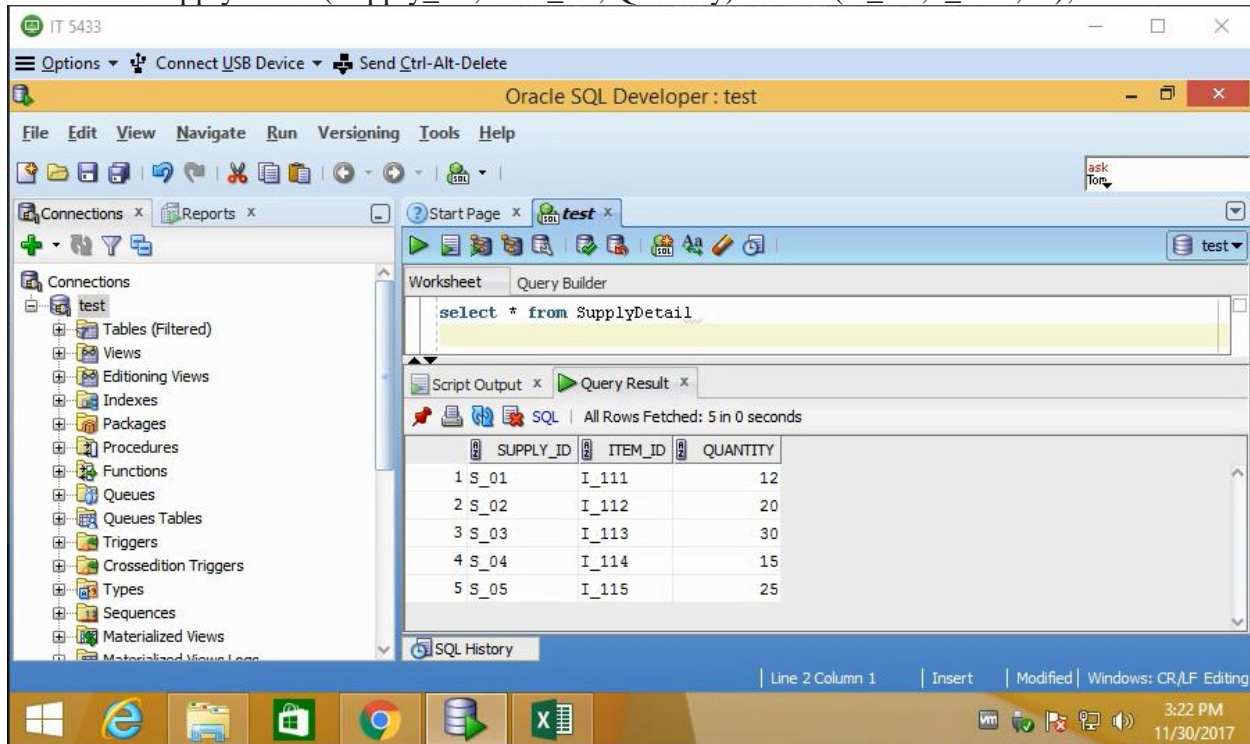
Insert into SupplyDetail (Supply_ID, Item_ID, Quantity) Values('S_01','I_111',12);

Insert into SupplyDetail (Supply_ID, Item_ID, Quantity) Values('S_02','I_112',20);

Insert into SupplyDetail (Supply_ID, Item_ID, Quantity) Values('S_03','I_113',30);

Insert into SupplyDetail (Supply_ID, Item_ID, Quantity) Values('S_04','I_114',15);

Insert into SupplyDetail (Supply_ID, Item_ID, Quantity) Values('S_05','I_115',25);



Item Table

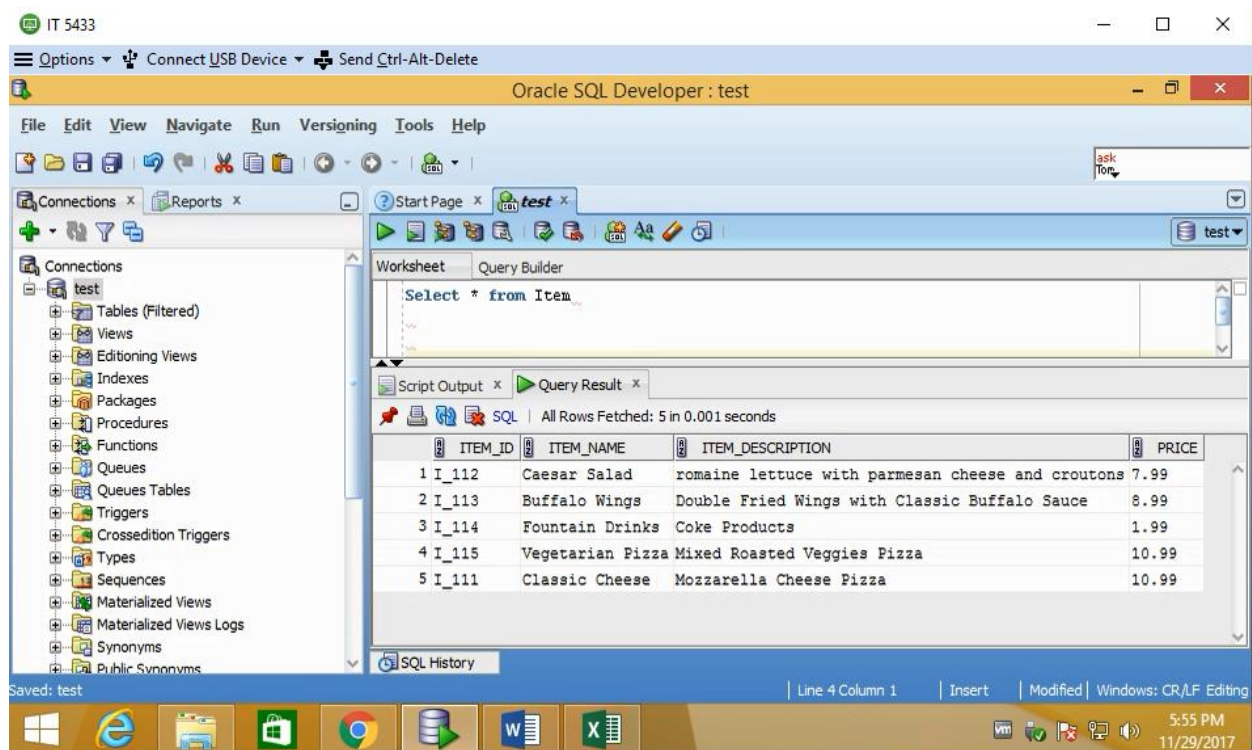
Insert into Item (Item_ID, Item_Name, Item_Description, Price) Values ('I_112','Caesar Salad', 'romaine lettuce with parmesan cheese and croutons','7.99');

Insert into Item (Item_ID, Item_Name, Item_Description, Price) Values ('I_113','Buffalo Wings', 'Double Fried Wings with Classic Buffalo Sauce','8.99');

Insert into Item (Item_ID, Item_Name, Item_Description, Price) Values ('I_114','Fountain Drinks', 'Coke Products','1.99');

Insert into Item (Item_ID, Item_Name, Item_Description, Price) Values ('I_115','Vegetarian Pizza', 'Mixed Roasted Veggies Pizza','10.99');

Insert into Item (Item_ID, Item_Name, Item_Description, Price) Values ('I_111','Classic Cheese', 'Mozzarella Cheese Pizza','10.99');



Supply Table

Insert into Supply (Supply_ID, Supply_Name, StockLow, Perishable) Values ('S_05','Flour','No','Yes');

Insert into Supply (Supply_ID, Supply_Name, StockLow, Perishable) Values ('S_03','Pizza Boxes', 'No', 'No');

Insert into Supply (Supply_ID, Supply_Name, StockLow, Perishable) Values ('S_02','Cutlery','Yes','No');

Insert into Supply (Supply_ID, Supply_Name, StockLow, Perishable) Values ('S_01','Vegetables','Yes','Yes');

Insert into Supply (Supply_ID, Supply_Name, StockLow, Perishable) Values ('S_04','Meat','No','Yes');

Insert into Supply (Supply_ID, Supply_Name, StockLow, Perishable) Values ('S_06','Cheese','Yes','Yes');

Insert into Supply (Supply_ID, Supply_Name, StockLow, Perishable) Values ('S_07','Paper Napkins', 'Yes', 'Yes');

Insert into Supply (Supply_ID, Supply_Name, StockLow, Perishable) Values ('S_08','Dry Spices','Yes','Yes');

The screenshot shows the Oracle SQL Developer interface. The main window displays a query result for the 'Supply' table. The query is 'Select * from Supply'. The result shows 8 rows of data. The columns are SUPPLY_ID, SUPPLY_NAME, STOCKLOW, and PERISHABLE. The status bar at the bottom indicates 'Line 1 Column 22 | Insert | Modified | Windows: CR/LF Editing' and the system clock shows '9:16 AM 12/1/2017'.

	SUPPLY_ID	SUPPLY_NAME	STOCKLOW	PERISHABLE
1	S_05	Flour	No	Yes
2	S_03	Pizza Boxes	No	No
3	S_02	Cutlery	Yes	No
4	S_01	Vegetables	Yes	Yes
5	S_04	Meat	No	Yes
6	S_06	Cheese	Yes	Yes
7	S_07	Paper Napkins	Yes	Yes
8	S_08	Dry Spices	Yes	Yes

Vendor Table

Insert into Vendor (Vendor_ID, Vendor_Name, Street, City, State, zip, Email_ID, Phone, info) values ('V_05','City Discount','587 Friends St', 'Jacksonville', 'Georgia', 66666, Friends@yum.com, 4564564567, 'Cleaning Supplies');

Insert into Vendor (Vendor_ID, Vendor_Name, Street, City, State, zip, Email_ID, Phone, info) values ('V_04', 'Kitchify','457 Nice St','Niceston','Georgia',22222,'Kitch@yum.com',4564564567,'Produce Vendor');

Insert into Vendor (Vendor_ID, Vendor_Name, Street, City, State, zip, Email_ID, Phone, info) values ('V_02', 'Angel Soft','123 Angel St','Angel','Georgia',55555,'Angel@yum.com',2342342345,'Cultery Vendor');

Insert into Vendor (Vendor_ID, Vendor_Name, Street, City, State, zip, Email_ID, Phone, info) values ('V_01', 'Sysco','458 Tide St','Atlanta','Georgia',99999,'Sysco@yum.com',1231231234,'Meat Vendor');

Insert into Vendor (Vendor_ID, Vendor_Name, Street, City, State, zip, Email_ID, Phone, info) values ('V_03', 'B and G Restaurant Supply','789 Holly St','Hover','Georgia',11111,'BG@yum.com',1231231234,'Kitchen Supplies');

Purchase Order Table

Insert into PurchaseOrder (PurchaseOrder_ID, PODate, Quantity, Amount, Vendor_ID, Supply_ID, Manager_ID) Values ('PO_01', (to_date('9/9/2017','DD/MM/YYYY')), 20, '2000.00', 'V_01','S_01','E_JS05');

Insert into PurchaseOrder (PurchaseOrder_ID, PODate, Quantity, Amount, Vendor_ID, Supply_ID, Manager_ID) Values ('PO_02', (to_date('30/11/2017','DD/MM/YYYY')),45,'1000.00', 'V_02','S_02','E_KW05');

Insert into PurchaseOrder (PurchaseOrder_ID, PODate, Quantity, Amount, Vendor_ID, Supply_ID, Manager_ID) Values ('PO_03', (to_date('11/02/2017','DD/MM/YYYY')),25,'4000.00', 'V_03','S_06','E_JS05');

Insert into PurchaseOrder (PurchaseOrder_ID, PODate, Quantity, Amount, Vendor_ID, Supply_ID, Manager_ID) Values ('PO_04', (to_date('13/11/2017','DD/MM/YYYY')),10,'3000.00', 'V_04','S_07','E_BP07');

Insert into PurchaseOrder (PurchaseOrder_ID, PODate, Quantity, Amount, Vendor_ID, Supply_ID, Manager_ID) Values ('PO_05', (to_date('28/11/2017','DD/MM/YYYY')),15,'500.00', 'V_05','S_08','E_JS05');

IT 5433

Options ▾ Connect USB Device ▾ Send Ctrl-Alt-Delete

Oracle SQL Developer : test

File Edit View Navigate Run Versing Tools Help

Start Page x test x

Connections test Cloud Connections

Worksheet Query Builder

Select * from PurchaseOrder

Script Output x Query Result x

SQL | All Rows Fetched: 5 in 0.004 seconds

	PURCHASEORDER_ID	PODATE	QUANTITY	AMOUNT	VENDOR_ID	SUPPLY_ID	MANAGER_ID
1	PO_01	09-SEP-17	20	2000	V_01	S_01	E_JS05
2	PO_02	30-NOV-17	45	1000	V_02	S_02	E_KW05
3	PO_03	11-FEB-17	25	4000	V_03	S_06	E_JS05
4	PO_04	13-NOV-17	10	3000	V_04	S_07	E_BP07
5	PO_05	28-NOV-17	15	500	V_05	S_08	E_JS05

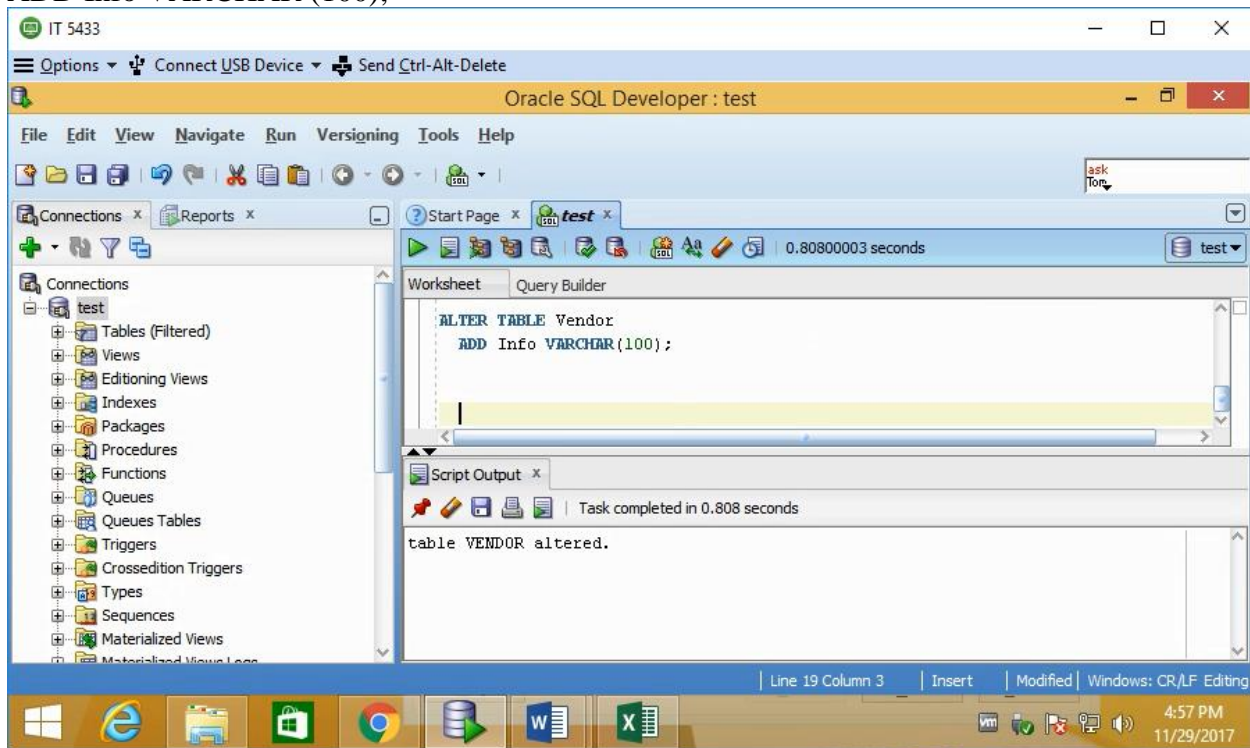
Line 1 Column 28 | Insert | Modified | Windows: CR/LF Editing

9:20 AM 12/1/2017

3.3 Queries

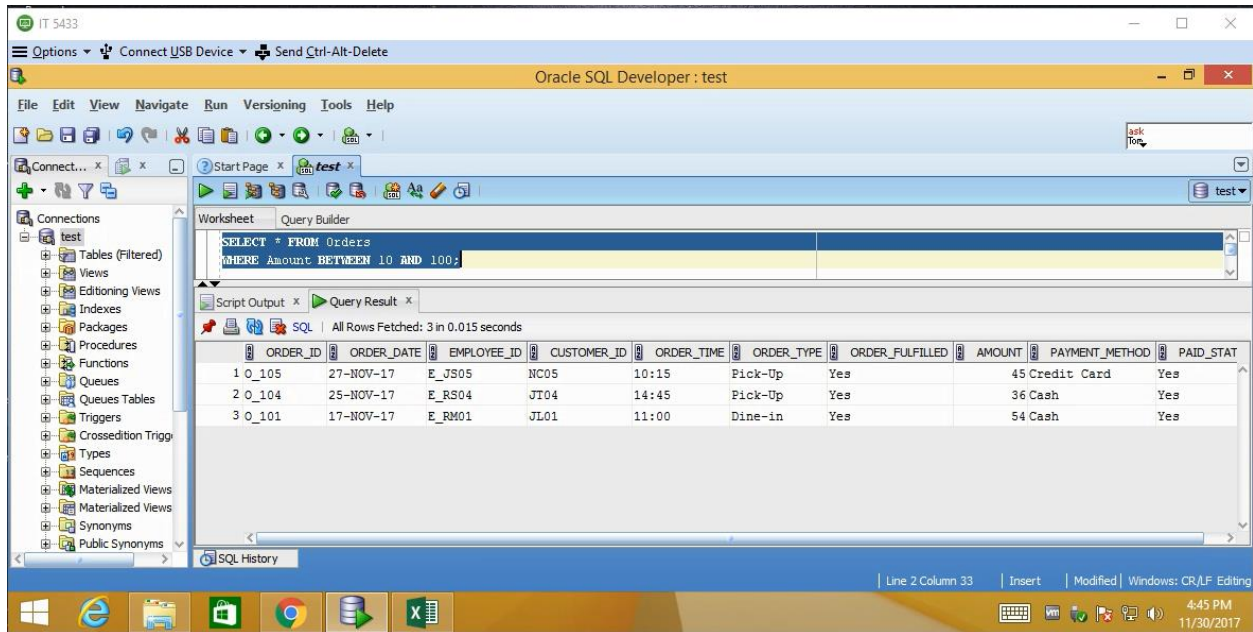
Alter Query

ALTER TABLE Vendor
ADD Info VARCHAR(100);



Between Query

SELECT * FROM Orders
WHERE Amount BETWEEN 10 AND 100;



The screenshot shows the Oracle SQL Developer interface. The main window displays the query: `SELECT * FROM Orders WHERE Amount BETWEEN 10 AND 100;`. The query result is shown in a table with 10 columns: ORDER_ID, ORDER_DATE, EMPLOYEE_ID, CUSTOMER_ID, ORDER_TIME, ORDER_TYPE, ORDER_FULFILLED, AMOUNT, PAYMENT_METHOD, and PAID_STAT. The result contains 3 rows of data.

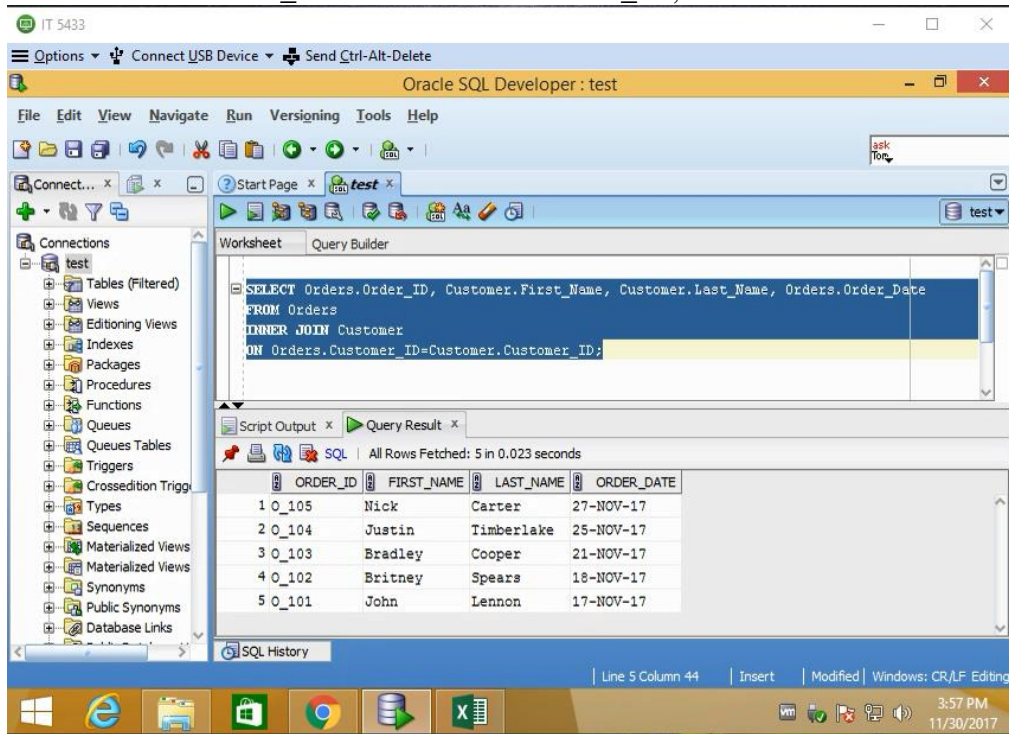
ORDER_ID	ORDER_DATE	EMPLOYEE_ID	CUSTOMER_ID	ORDER_TIME	ORDER_TYPE	ORDER_FULFILLED	AMOUNT	PAYMENT_METHOD	PAID_STAT
1 O_105	27-NOV-17	E_JS05	NC05	10:15	Pick-Up	Yes	45	Credit Card	Yes
2 O_104	25-NOV-17	E_RS04	JT04	14:45	Pick-Up	Yes	36	Cash	Yes
3 O_101	17-NOV-17	E_RM01	JL01	11:00	Dine-in	Yes	54	Cash	Yes

Inner Join Query

SELECT Orders. Order_ID, Customer.First_Name, Customer.Last_Name, Orders.Order_Date
From Orders

INNER JOIN Customer

ON Orders.Customer_ID = Customer.Customer_ID;



The screenshot shows the Oracle SQL Developer interface. The main window displays a query in the Worksheet tab:

```
SELECT Orders.Order_ID, Customer.First_Name, Customer.Last_Name, Orders.Order_Date
FROM Orders
INNER JOIN Customer
ON Orders.Customer_ID=Customer.Customer_ID;
```

Below the query, the Query Results pane shows the output of the query. It indicates that all rows were fetched in 0.023 seconds. The results are displayed in a table with the following columns: ORDER_ID, FIRST_NAME, LAST_NAME, and ORDER_DATE.

ORDER_ID	FIRST_NAME	LAST_NAME	ORDER_DATE
10_105	Nick	Carter	27-NOV-17
20_104	Justin	Timberlake	25-NOV-17
30_103	Bradley	Cooper	21-NOV-17
40_102	Britney	Spears	18-NOV-17
50_101	John	Lennon	17-NOV-17

The bottom status bar shows the current line and column (Line 5 Column 44) and the date/time (3:57 PM 11/30/2017).

Full Outer Join Query

```
SELECT Customer.First_Name, Customer.Last_Name, Orders.Order_ID  
FROM Customer  
FULL OUTER JOIN Orders ON Customer.Customer_ID = Orders.Customer_ID  
ORDER BY Customer.First_Name;
```

The screenshot displays the Oracle SQL Developer interface. The main window shows a query in the 'Query Builder' tab. The query is a Full Outer Join between the 'Customer' and 'Orders' tables, ordered by 'Customer.First_Name'. The 'Connections' pane on the left shows the 'test' connection. The 'Query Result' pane at the bottom displays the results of the query, showing 5 rows of data.

	FIRST_NAME	LAST_NAME	ORDER_ID
1	Bradley	Cooper	O_103
2	Britney	Spears	O_102
3	John	Lennon	O_101
4	Justin	Timberlake	O_104
5	Nick	Carter	O_105

Left Join Query

```
SELECT Customer.First_Name, Customer.Last_Name, Orders.Order_ID  
FROM Customer  
LEFT JOIN Orders ON Customer.Customer_ID = Orders.Customer_ID  
ORDER BY Customer.First_Name;
```

The screenshot displays the Oracle SQL Developer interface. The main window shows a worksheet with the following SQL query:

```
SELECT Customer.First_Name, Customer.Last_Name, Orders.Order_ID  
FROM Customer  
LEFT JOIN Orders ON Customer.Customer_ID = Orders.Customer_ID  
ORDER BY Customer.First_Name;
```

Below the query editor, the 'Query Result' tab is active, showing the results of the query. The status bar indicates 'All Rows Fetched: 5 in 0.056 seconds'. The results are displayed in a table with the following columns: FIRST_NAME, LAST_NAME, and ORDER_ID.

	FIRST_NAME	LAST_NAME	ORDER_ID
1	Bradley	Cooper	O_103
2	Britney	Spears	O_102
3	John	Lennon	O_101
4	Justin	Timberlake	O_104
5	Nick	Carter	O_105

The bottom of the screen shows the Windows taskbar with various application icons and the system clock indicating 4:47 PM on 11/30/2017.

Right Join Query

```
SELECT Orders.Order_ID, Employee.Last_Name, Employee.First_Name  
FROM Orders  
RIGHT JOIN Employee ON Orders.Employee_ID = Employee.Employee_ID  
ORDER BY Orders.Order_ID;
```

The screenshot displays the Oracle SQL Developer interface. The 'Connections' pane on the left shows a connection named 'test'. The 'Query Builder' pane in the center contains the following SQL query:

```
SELECT Orders.Order_ID, Employee.Last_Name, Employee.First_Name  
FROM Orders  
RIGHT JOIN Employee ON Orders.Employee_ID = Employee.Employee_ID  
ORDER BY Orders.Order_ID;
```

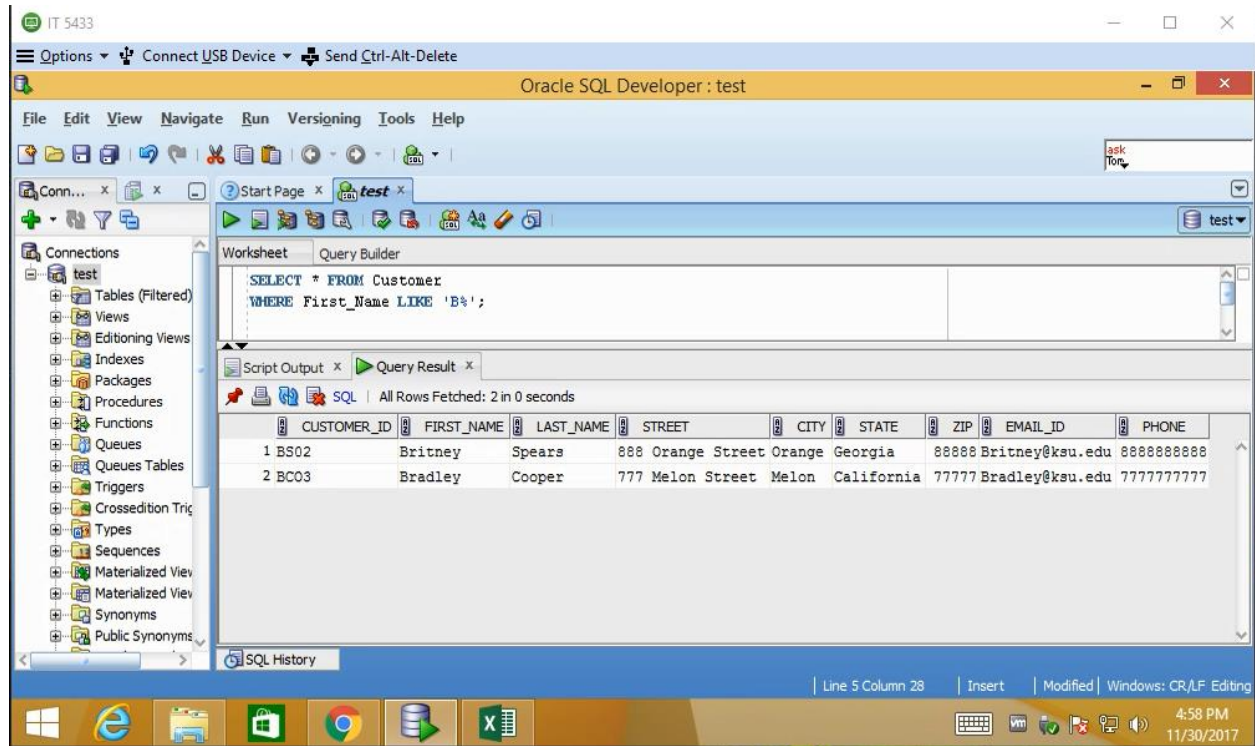
The 'Query Result' pane below the query shows the results of the query. It indicates that all rows were fetched in 0.004 seconds. The results are displayed in a table with three columns: ORDER_ID, LAST_NAME, and FIRST_NAME.

ORDER_ID	LAST_NAME	FIRST_NAME
1 (null)	Pitt	Brad
2 (null)	Martin	Ricky
3 (null)	West	Kayne
4 (null)	Cox	Courtney
5 (null)	Hall	Pheobe
6 (null)	Austin	Jane
7 (null)	Duff	Hilary
8 (null)	Stevens	Ren
9 (null)	Simpsons	Jessica
10 (null)	Montana	Hanna

The bottom status bar shows the current line and column (Line 4 Column 27) and the date and time (9:15 AM 12/2/2017).

Like Query

```
SELECT * FROM Customer  
WHERE First_Name LIKE 'B%';
```



The screenshot displays the Oracle SQL Developer interface. The main window shows a query in the 'Query Builder' tab: `SELECT * FROM Customer WHERE First_Name LIKE 'B%';`. The 'Query Result' tab below shows the execution results, indicating 'All Rows Fetched: 2 in 0 seconds'. The results are presented in a table with the following columns: CUSTOMER_ID, FIRST_NAME, LAST_NAME, STREET, CITY, STATE, ZIP, EMAIL_ID, and PHONE. Two rows are displayed: one for Britney Spears and one for Bradley Cooper.

CUSTOMER_ID	FIRST_NAME	LAST_NAME	STREET	CITY	STATE	ZIP	EMAIL_ID	PHONE
1 BS02	Britney	Spears	888 Orange Street	Orange	Georgia	88888	Britney@ksu.edu	8888888888
2 BC03	Bradley	Cooper	777 Melon Street	Melon	California	77777	Bradley@ksu.edu	7777777777

Union Query

SELECT City FROM Employee UNION
SELECT City FROM Vendor ORDER BY City;

The screenshot displays the Oracle SQL Developer interface. The main window is titled "Oracle SQL Developer : test". The left pane shows the "Connections" tree with a "test" connection selected. The central pane is in "Query Builder" mode, showing the following SQL query:

```
SELECT City FROM Employee UNION  
SELECT City FROM Vendor ORDER BY City;
```

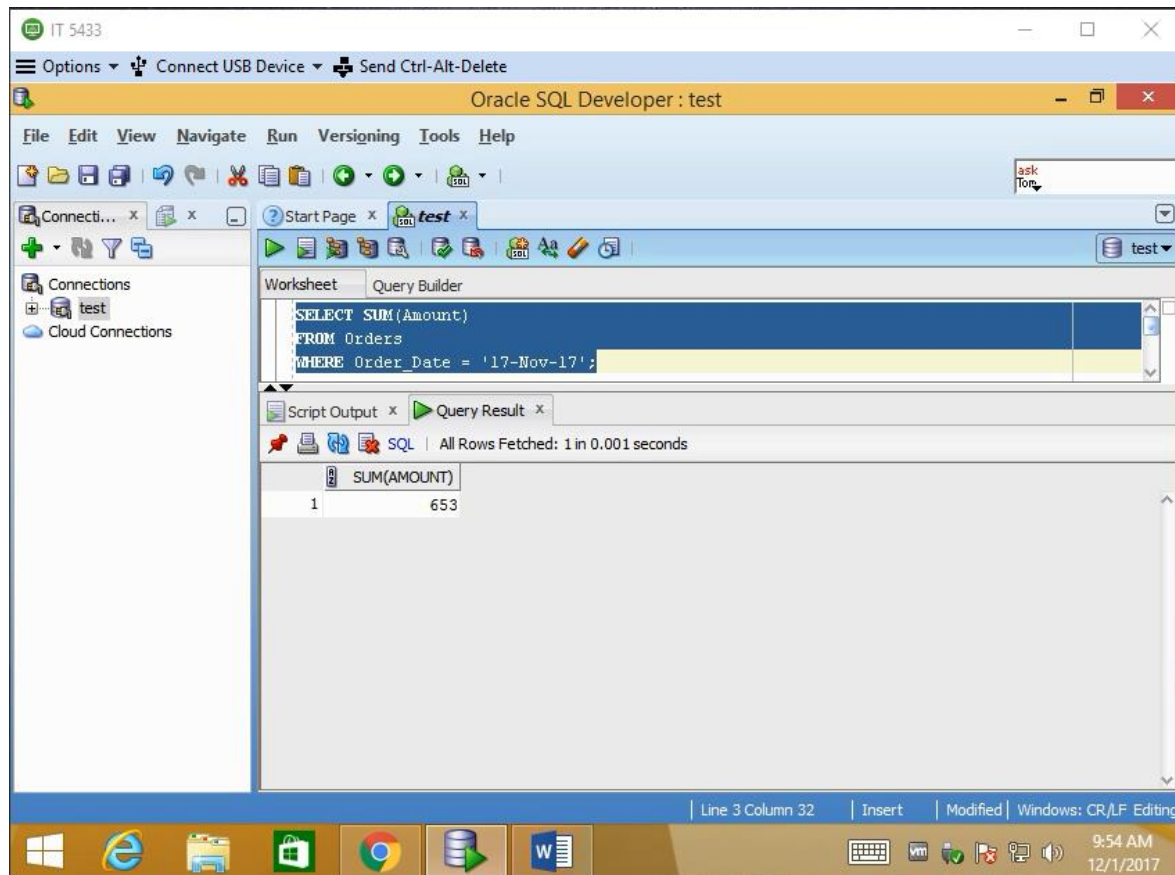
Below the query, the "Query Result" pane shows the execution status: "All Rows Fetched: 9 in 0.051 seconds". The results are displayed in a table with one column, "CITY", and nine rows:

CITY
1 Atlanta
2 Central
3 Dakota
4 Holler
5 Merlot
6 Middleton
7 Peachtree
8 Prescott
9 Suwanee

The bottom status bar indicates "Line 3 Column 1", "Insert", "Modified", and "Windows: CR/LF Editing". The system tray at the bottom shows the date and time: "9:11 AM 12/2/2017".

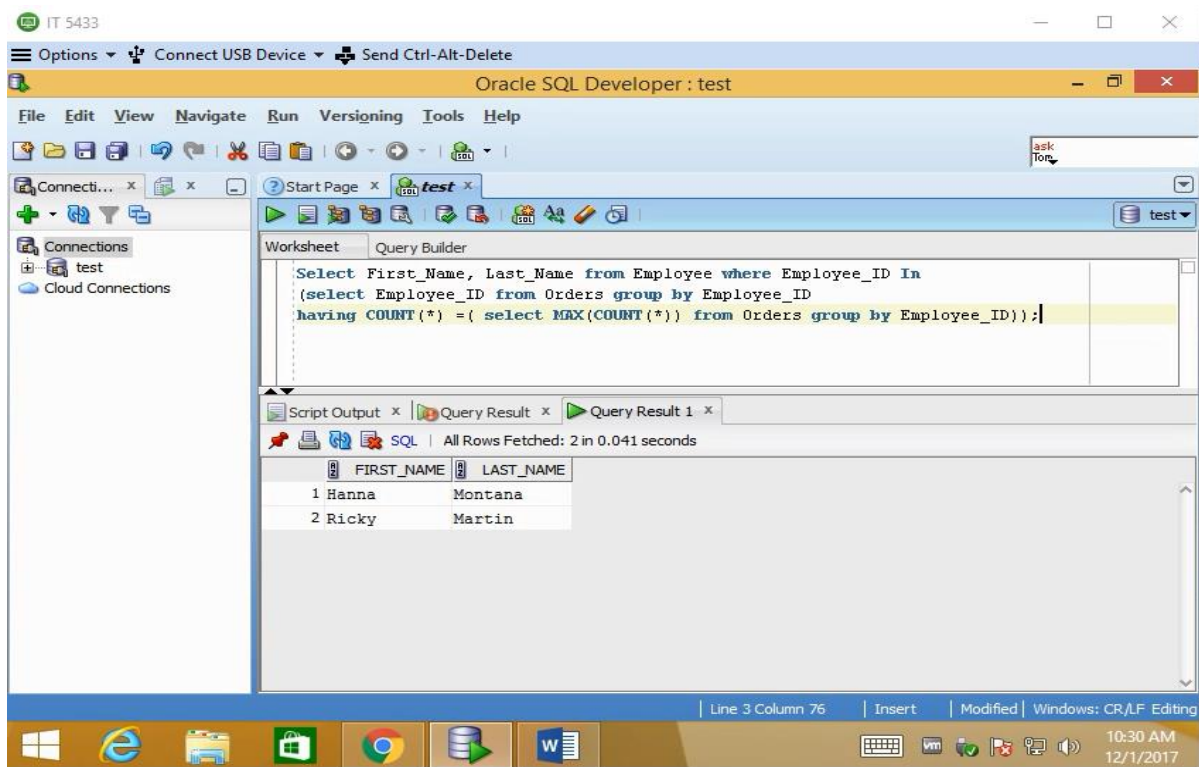
Total Sales for a day

```
SELECT SUM(Amount) FROM Orders  
WHERE Order_Date = '17-Nov-17';
```



Employee with Maximum Orders

Select First_Name, Last_Name from Employee where Employee_ID In (Select Employee_ID from Orders group by Employee_ID having COUNT (*) = (select Max (COUNT (*)) from Orders group by Employee_ID));



The screenshot shows the Oracle SQL Developer interface. The main window displays a query in the Worksheet tab:

```
Select First_Name, Last_Name from Employee where Employee_ID In  
(select Employee_ID from Orders group by Employee_ID  
having COUNT (*) = ( select MAX(COUNT (*)) from Orders group by Employee_ID));
```

Below the query, the Query Result tab shows the results of the query. The status bar indicates "All Rows Fetched: 2 in 0.041 seconds". The results are displayed in a table with two columns: FIRST_NAME and LAST_NAME.

	FIRST_NAME	LAST_NAME
1	Hanna	Montana
2	Ricky	Martin

The bottom of the screen shows the Windows taskbar with various application icons and the system clock displaying 10:30 AM on 12/1/2017.

Orders for a Day

Select * FROM Orders

WHERE ((TO_CHAR(Order_Date, 'YYYY-Mon-DD')) = '2017-Nov-17');

The screenshot shows the Oracle SQL Developer interface. The 'Connections' pane on the left lists various database objects. The 'Worksheet' pane displays the following SQL query:

```
SELECT *  
FROM Orders  
WHERE ((TO_CHAR(Order_date, 'YYYY-Mon-DD')) = '2017-Nov-17');
```

The 'Query Result' pane shows the output of the query, which consists of 6 rows. The status bar indicates 'All Rows Fetched: 6 in 0.005 seconds'.

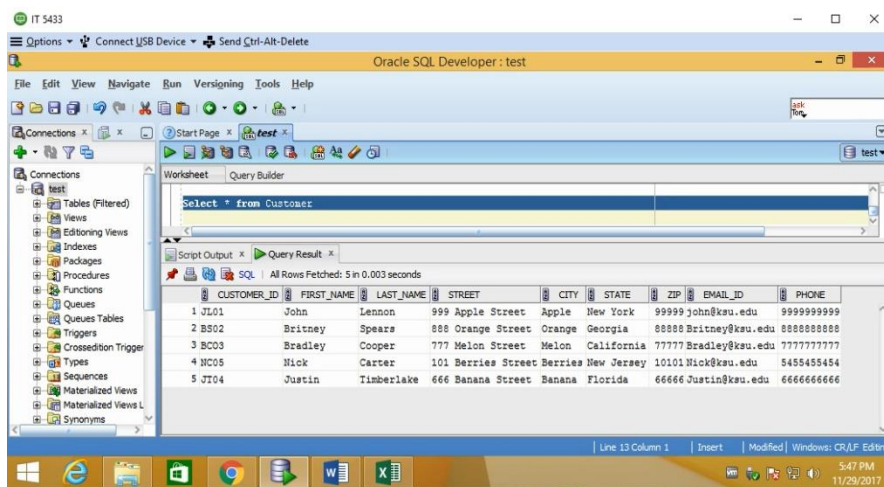
ORDER_ID	ORDER_DATE	EMPLOYEE_ID	CUSTOMER_ID	ORDER_TIME	ORDER_TYPE
1 O_101	17-NOV-17	E_RM01	JL01	11:00	Dine-in
2 O_106	17-NOV-17	E_RM01	JL01	8:00	Dine-in
3 O_109	17-NOV-17	E_RM02	JT04	14:00	Pick-Up
4 O_110	17-NOV-17	E_RM01	BS02	14:00	Take-Out
5 O_108	17-NOV-17	E_HD03	NC05	14:00	Take-Out
6 O_107	17-NOV-17	E_RM02	BS02	14:00	Take-Out

The bottom status bar shows 'Line 1 Column 1 | Insert | Modified | Windows: CR/LF Editing' and the system clock displays '9:30 AM 12/2/2017'.

3.4 Workflow Demonstrations

1. New customer comes to the restaurant

Insert into Customer (Customer_ID, First_Name, Last_Name, Street, City, State , Zip, Email_ID, Phone) values ('NC05','Nick','Carter','101 Berries Street','Berries','New Jersey',10101,'Nick@ksu.edu','5455455454');

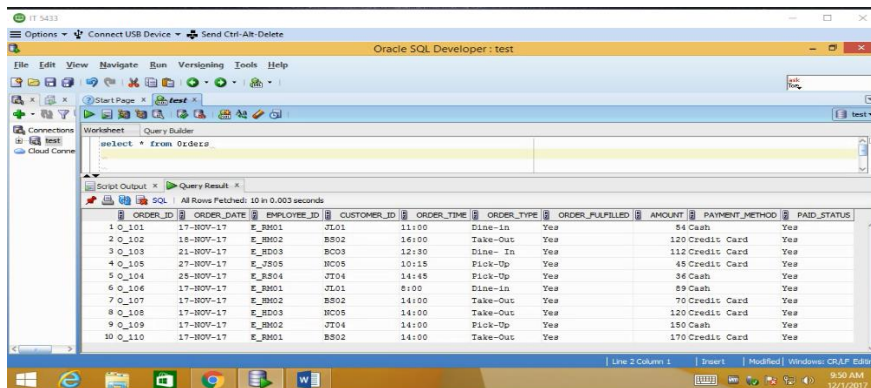


The screenshot shows the Oracle SQL Developer interface. The 'Query Result' window displays the following data:

CUSTOMER_ID	FIRST_NAME	LAST_NAME	STREET	CITY	STATE	ZIP	EMAIL_ID	PHONE
1 J101	John	Lennon	999 Apple Street	Apple	New York	99999	john@ksu.edu	9999999999
2 BS02	Britney	Spears	888 Orange Street	Orange	Georgia	88888	Britney@ksu.edu	8888888888
3 BC03	Bradley	Cooper	777 Melon Street	Melon	California	77777	Bradley@ksu.edu	7777777777
4 NC05	Nick	Carter	101 Berries Street	Berries	New Jersey	10101	Nick@ksu.edu	5455455454
5 J104	Justin	Timberlake	666 Banana Street	Banana	Florida	66666	Justin@ksu.edu	6666666666

2. Places order

Insert into Orders (Order_ID, Order_Date, Employee_ID, Customer_ID, Order_Time, Order_Type, Order_Fulfilled, Amount, Payment_Method, Paid_Status) values ('O_108', (to_date('17/11/2017','DD/MM/YYYY')), 'E_HD03', 'NC05','14:00','Take-Out','Yes',120,'Credit Card','Yes');

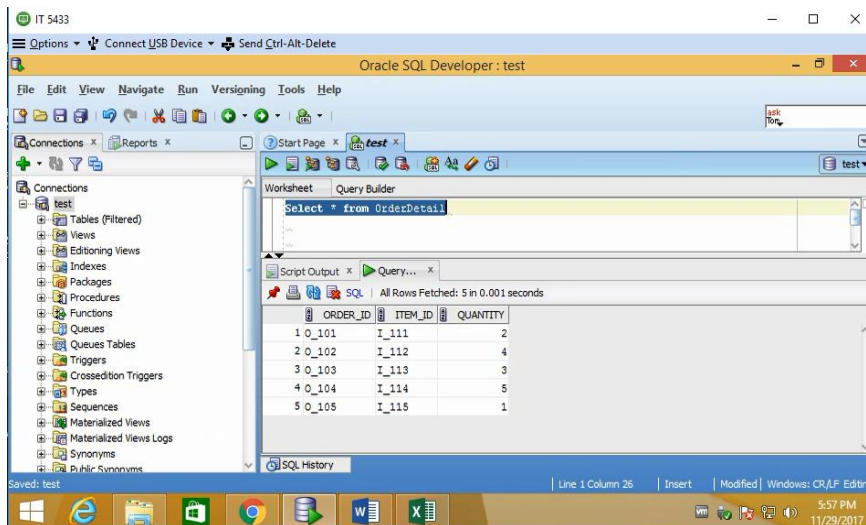


The screenshot shows the Oracle SQL Developer interface. The 'Query Result' window displays the following data:

ORDER_ID	ORDER_DATE	EMPLOYEE_ID	CUSTOMER_ID	ORDER_TIME	ORDER_TYPE	ORDER_FULFILLED	AMOUNT	PAYMENT_METHOD	PAID_STATUS
1 O_101	17-NOV-17	E_RH01	J101	11:00	Dine-In	Yes	94	Cash	Yes
2 O_102	18-NOV-17	E_RH02	BS02	16:00	Take-Out	Yes	120	Credit Card	Yes
3 O_103	21-NOV-17	E_RH03	BC03	12:30	Dine-In	Yes	112	Credit Card	Yes
4 O_105	27-NOV-17	E_RH05	NC05	10:15	Pick-Up	Yes	45	Credit Card	Yes
5 O_104	28-NOV-17	E_RH04	J104	14:45	Pick-Up	Yes	36	Cash	Yes
6 O_106	17-NOV-17	E_RH01	J101	8:00	Dine-In	Yes	89	Cash	Yes
7 O_107	17-NOV-17	E_RH02	BS02	14:00	Take-Out	Yes	70	Credit Card	Yes
8 O_108	17-NOV-17	E_RH03	NC05	14:00	Take-Out	Yes	120	Credit Card	Yes
9 O_109	17-NOV-17	E_RH02	J104	14:00	Pick-Up	Yes	150	Cash	Yes
10 O_110	17-NOV-17	E_RH01	BS02	14:00	Take-Out	Yes	170	Credit Card	Yes

3. Staff employee places the order

Insert into OrderDetail (Order_ID, Item_ID, Quantity) values ('O_105', 'I_115',1);



The screenshot shows the Oracle SQL Developer interface. The 'Script Output' window displays the results of a query: 'Select * from OrderDetail;'. The results are as follows:

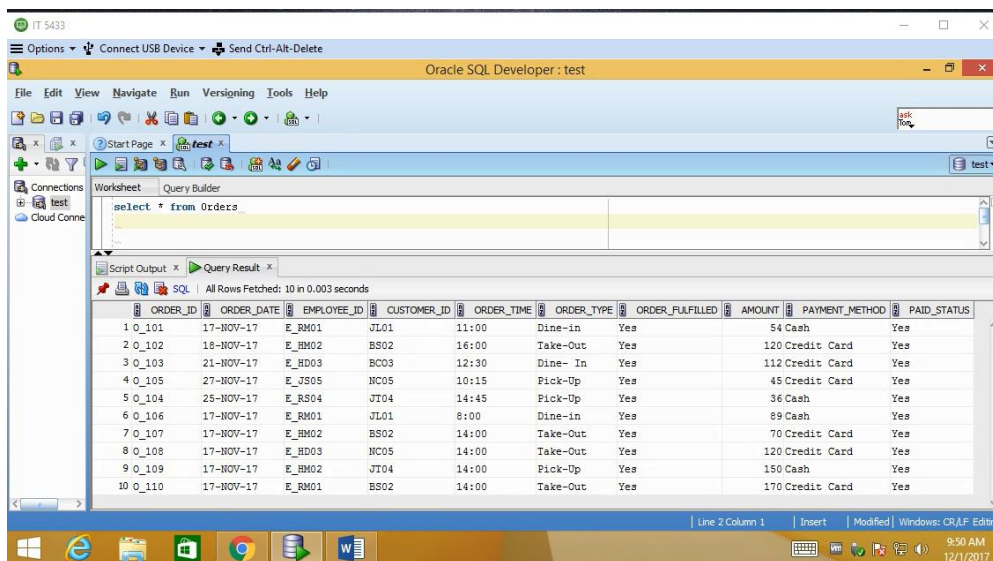
ORDER_ID	ITEM_ID	QUANTITY
1 O_101	I_111	2
2 O_102	I_112	4
3 O_103	I_113	3
4 O_104	I_114	5
5 O_105	I_115	1

4. Updates the order to order fulfilled

UPDATE Orders

SET Order_Fulfilled = 'Yes'

WHERE Order_ID = '105';

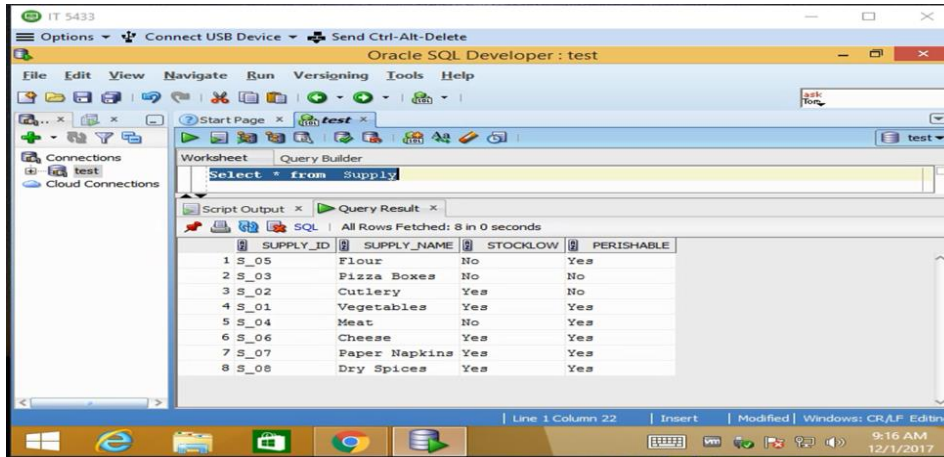


The screenshot shows the Oracle SQL Developer interface. The 'Script Output' window displays the results of a query: 'select * from Orders'. The results are as follows:

ORDER_ID	ORDER_DATE	EMPLOYEE_ID	CUSTOMER_ID	ORDER_TIME	ORDER_TYPE	ORDER_FULFILLED	AMOUNT	PAYMENT_METHOD	PAID_STATUS
1 O_101	17-NOV-17	E_RM01	JL01	11:00	Dine-in	Yes	54	Cash	Yes
2 O_102	18-NOV-17	E_RM02	BS02	16:00	Take-Out	Yes	120	Credit Card	Yes
3 O_103	21-NOV-17	E_HD03	BC03	12:30	Dine-In	Yes	112	Credit Card	Yes
4 O_105	27-NOV-17	E_US05	NC05	10:15	Pick-Up	Yes	45	Credit Card	Yes
5 O_104	25-NOV-17	E_RS04	JT04	14:45	Pick-Up	Yes	36	Cash	Yes
6 O_106	17-NOV-17	E_RM01	JL01	8:00	Dine-in	Yes	89	Cash	Yes
7 O_107	17-NOV-17	E_RM02	BS02	14:00	Take-Out	Yes	70	Credit Card	Yes
8 O_108	17-NOV-17	E_HD03	NC05	14:00	Take-Out	Yes	120	Credit Card	Yes
9 O_109	17-NOV-17	E_RM02	JT04	14:00	Pick-Up	Yes	150	Cash	Yes
10 O_110	17-NOV-17	E_RM01	BS02	14:00	Take-Out	Yes	170	Credit Card	Yes

5. Checks inventory and updates if supply table StockLow.

```
SELECT *  
FROM Supply
```

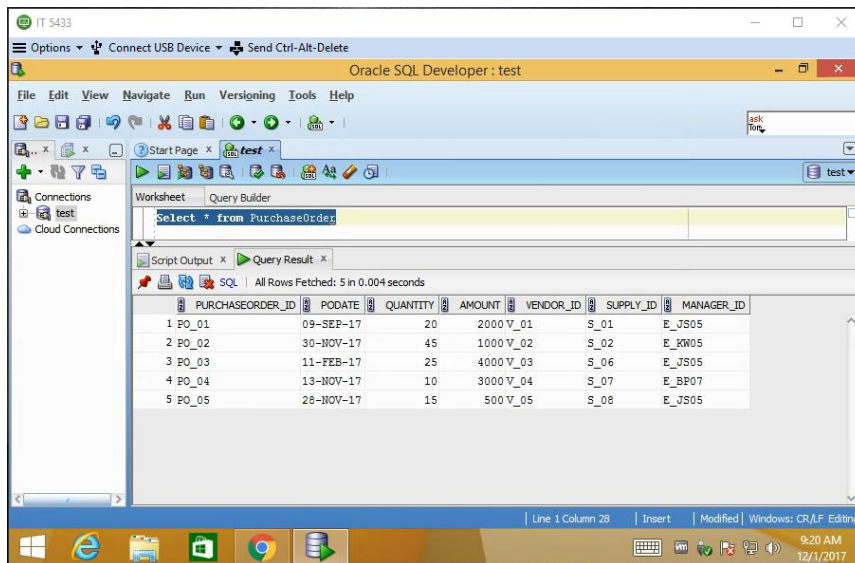


The screenshot shows the Oracle SQL Developer interface. The 'Query Result' window displays the following data:

SUPPLY_ID	SUPPLY_NAME	STOCKLOW	PERISHABLE
1 S_05	Flour	No	Yes
2 S_03	Pizza Boxes	No	No
3 S_02	Cutlery	Yes	No
4 S_01	Vegetables	Yes	Yes
5 S_04	Meat	No	Yes
6 S_06	Cheese	Yes	Yes
7 S_07	Paper Napkins	Yes	Yes
8 S_08	Dry Spices	Yes	Yes

6. Manager places purchase order if stock is low in the supply table.

```
Insert into PurchaseOrder (PurchaseOrder_ID, PODate, Quantity, Amount, Vendor_ID,  
Supply_ID, Manager_ID) Values ('PO_05', (to_date('28/11/2017','DD/MM/YYYY')),15,'500.00',  
'V_05','S_08','E_JS05');
```



The screenshot shows the Oracle SQL Developer interface. The 'Query Result' window displays the following data:

PURCHASEORDER_ID	PODATE	QUANTITY	AMOUNT	VENDOR_ID	SUPPLY_ID	MANAGER_ID
1 PO_01	09-SEP-17	20	2000	V_01	S_01	E_JS05
2 PO_02	30-NOV-17	45	1000	V_02	S_02	E_KW05
3 PO_03	11-FEB-17	25	4000	V_03	S_06	E_JS05
4 PO_04	13-NOV-17	10	3000	V_04	S_07	E_BP07
5 PO_05	28-NOV-17	15	500	V_05	S_08	E_JS05