

**CERTIFICATE OF SPECIFIC LEARNING DIFFICULTY**

**…………………….**

**Student ID number:** 1008015

**Student:** Mohammod Habibur Rahman

This certificate must be copied and put on the front of all assignments and examinations.

This Certificate is for your use only. Any modification or misuse is a breach of university regulations.

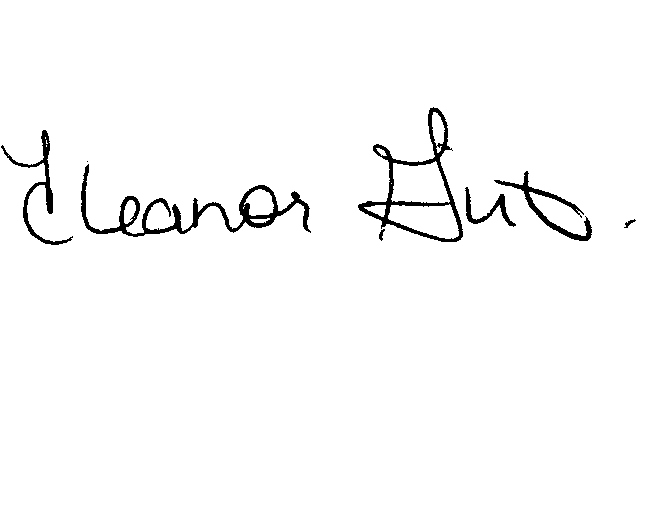
**Assessors and Examiners:**

This certifies that this student has been assessed as a person with specific learning difficulties. In marking, try to look beyond the poor language skills for knowledge, content and ideas, **unless written language and communication skills are essential learning outcomes.**

**Students must be fully aware that any adjustments do not override prescribed professional standards required in their course.**

**For further information on marking guidelines please look on the DDAC website:**

<http://www.uel.ac.uk/studentservices/supportingyou/staff/index.htm>

**Signed:** 

**Name: Eleanor Girt, Head of Disability/Dyslexia and Access Centre**

**Date: 04 December 2012**



**Advanced Database Development**

**Module code: SD3042**

**Institution**: University Of East London (UEL)

Mohammod Habibur Rahman

**ID**: u1008015

Alexandros Akrivopoulos-Hughes

**ID:** u1235441

**Module Leader: Juliette Alfred-Lewars**

Index

Table of Contents

[1.0 Case Study 4](#_Toc341098570)

[1.1 Introduction 4](#_Toc341098571)

[1.3 ER1 5](#_Toc341098573)

[1.4 ER2](#_Toc341098574) 8

[1.5 Table Structure, Details, Creation, “Population” and Constraint Scripts](#_Toc341098575) 9

[1.6 PL/SQL](#_Toc341098576) 38

[1.7 Showing the Queries](#_Toc341098578) 41

[1.8 Group Member Participation](#_Toc341098579) 43

[2.0 Evaluation](#_Toc341098580) 44

[2.1 Recommendation](#_Toc341098581) 44

[2.2 References](#_Toc341098582) 45

# 1.0 Case Study

In this assessment we have been told that a company called Natural Health Shop wants us to develop them a booking system for the company which was implemented as a relational database which contains some business logic functionality, as they were having problems making orders and were also double booking rooms. Therefore this company had approached us in good faith and has asked us to design and implement for them, a database. They don’t want the whole system to take over the shop, but allow the employees not to double book rooms and also they can have a system where they can see how the exact condition of the stock so they don’t need to order more than what is needed. Therefore we, as a team will extract the business rules of the company and present them a design as well as a demo implementation of the system.

# 1.1 Introduction

Our group will design and develop a database for a Natural Health Shop to help the staff to run the shop more efficiently.

In order to develop a useful system we will have to fully understand and keep in mind the business rules of the company, create an ER1, a logical model, where we can define the entities, attributes and relationships between them. After that the flow of work will proceed by converting the logical model to a relational model etc.

The DBMS (database management system), must be able to cope with some of the questions which are important to the business.

We will be using Oracle DBMS as the platform to develop our database management system for the company.

Before we start developing this database management system we must think of what each person must do. Once everybody has been assigned his part of work, we will then start to build the DBMS one part at a time. We will also make sure that when we are creating the DBMS we are taking into consideration that the DBMS can carry out some simple tasks which the business needs from the system. Oracle will be used in this case to provide the database management system (DBMS) which will help us into programming the business logic using PL/SQL.

We as a group will be working hard into developing consistent, scalable and efficient system to help the staff to keep the business running. We will also create some close to reality sample data in order to run business scenarios and investigate if the system can really respond to the company’s needs.

# 1.3 ER1

This is the ER1 which we developed and demonstrates the logical model of the company.

Figure : ER1

The relationships between entities can be fully demonstrated using 1..\* (one to many), \*..\*(Many to Many) or 1..1(One to One) relationships.

If we forward engineer the logical model appearing in ERD1, along with some modifications, we will get the ERD2.

**A brief description of the relationships is given below:**

1. Each **Product** *is supplied* by many **Suppliers**  
   Each **Supplier** *supplies* many **Products**
2. Each **Supplier** *supports* many **Shops**  
   Each **Shop** *is supported* by many **Suppliers**
3. Each **Product** *exists* in many **ShopOrders**  
   Each **ShopOrder** *contains* many **Products**
4. Each **ShopOrder** *belongs* to one **Shop**  
   Each **Shop** *contains* many **ShopOrders**.
5. Each **Product** *has* many **Stocks**  
   Each **Stock** *contains* one **Product**
6. Each **Stock** *belongs* to one **Shop**  
   Each **Shop** *contains* many **Stock items**
7. Each **Stock** *is used* by many **Inventory items**  
   Each **Inventory** *uses* one **Stock item**
8. Each **Inventory** *is used* by one **Treatment**  
   Each **Treatment** *uses* many **Inventory items**
9. Each **Shop** *contains* many **Rooms**  
   Each **Room** *is located* at one **Shop**
10. Each **Shop** *has* many **Employees**  
    Each **Employee** *works* in one **Shop**
11. Each **Shop** *manages* many **Employees**  
    Each **Employee** *is managed* by one **Shop**  
    Each **Shop** *is managed* by one **Employee**
12. Each **Employee** possesses many **Specialties**  
    Each **Specialty** can be possessed by many **Employees**
13. Each **Employee** *serves* on many **Bookings**  
    Each **Booking** *is served* by one **Employee**
14. Each **Room** *is booked* by many **Bookings**  
    Each **Booking** *uses* one **Room**
15. Each **Treatment** *exists* in many **Bookings**  
    Each **Booking** *operates* one **Treatment**
16. Each **Customer** *is served* in many **Bookings**  
    Each **Booking** *is booked* by one **Customer**

**Except from the relationship descriptions that include our assumptions as well as our client’s answers to our questions, we should also state that bookings can be only made at an exact hour, like 2 o’clock, not 2:15, in order for the booking system to behave smoothly.**

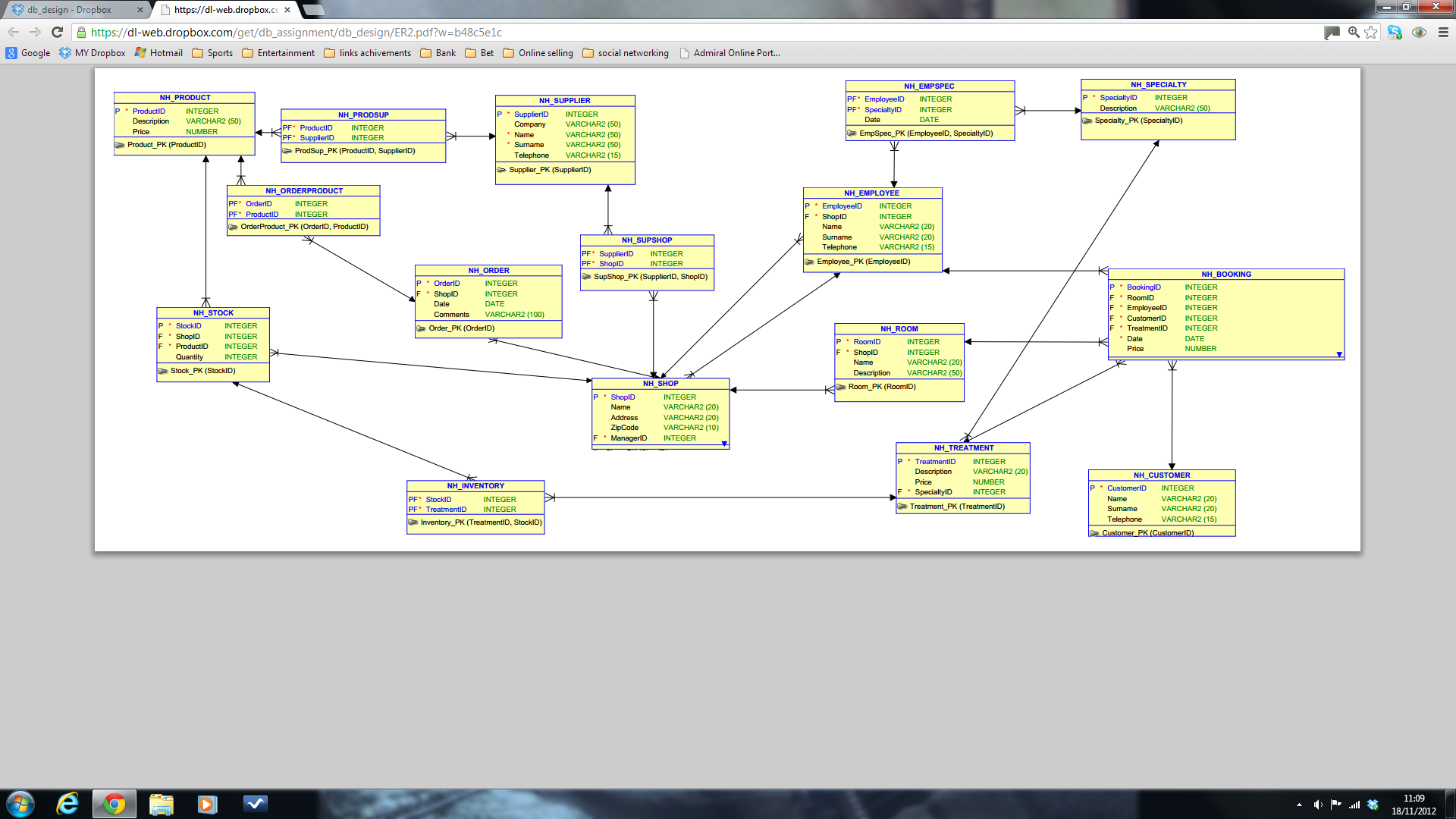
## 1.2 Database Modelling and Implementation:

**Main entities of Natural Health Shop**

* Product
* Supplier
* ShopOrder
* Stock
* Shop
* Inventory
* Employee
* Room
* Treatment
* Speciality
* Booking
* Customers

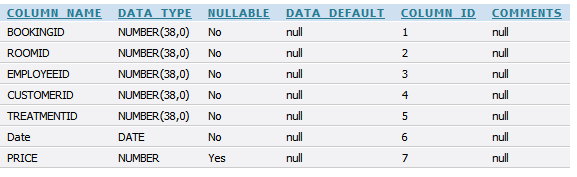
*These are our 12 entities for the company which we think they are needed after reading the company’s requirements.*

# 1.4 ER2

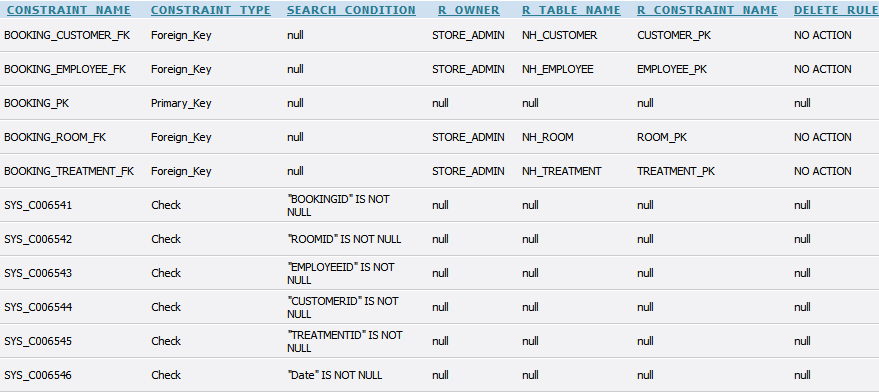


# 1.5 Listing of all the tables

**NH\_BOOKING Table Structure**



**NH\_BOOKING Table Constraints**



**NH\_BOOKING Creation, Population and Constraint Script**

CREATE TABLE NH\_BOOKING

(

BookingID INTEGER NOT NULL ,

RoomID INTEGER NOT NULL ,

EmployeeID INTEGER NOT NULL ,

CustomerID INTEGER NOT NULL ,

TreatmentID INTEGER NOT NULL ,

"Date" DATE NOT NULL ,

Price NUMBER

)

;

ALTER TABLE NH\_BOOKING

ADD CONSTRAINT Booking\_PK PRIMARY KEY ( BookingID ) ;

/\* BookingID INT, RoomID INT, EmployeeID INT, CustomerID INT, TreatmentID INT, Date DATE, Price DECIMAL \*/

INSERT INTO NH\_BOOKING VALUES

(1, 1, 1, 1, 1, to\_date('04/07/2010:12:00:00PM', 'dd/mm/yyyy:hh:mi:ssam'), 50.2);

INSERT INTO NH\_BOOKING VALUES

(2, 11, 12, 14, 11, to\_date('04/07/2010:01:00:00PM', 'dd/mm/yyyy:hh:mi:ssam'), 45);

INSERT INTO NH\_BOOKING VALUES

(3, 4, 2, 4, 6, to\_date('04/07/2010:02:00:00PM', 'dd/mm/yyyy:hh:mi:ssam'), 70);

INSERT INTO NH\_BOOKING VALUES

(4, 7, 4, 6, 8, to\_date('04/07/2010:03:00:00PM', 'dd/mm/yyyy:hh:mi:ssam'), 85);

INSERT INTO NH\_BOOKING VALUES

(5, 2, 10, 5, 9, to\_date('05/07/2010:12:00:00PM', 'dd/mm/yyyy:hh:mi:ssam'), 90);

INSERT INTO NH\_BOOKING VALUES

(6, 4, 9, 3, 5, to\_date('06/07/2010:12:00:00PM', 'dd/mm/yyyy:hh:mi:ssam'), 54);

INSERT INTO NH\_BOOKING VALUES

(7, 12, 5, 7, 3, to\_date('08/07/2010:12:00:00PM', 'dd/mm/yyyy:hh:mi:ssam'), 44);

INSERT INTO NH\_BOOKING VALUES

(8, 13, 3, 8, 6, to\_date('25/09/2011:01:00:00PM', 'dd/mm/yyyy:hh:mi:ssam'), 40);

INSERT INTO NH\_BOOKING VALUES

(9, 11, 1, 2, 12, to\_date('19/07/2012:05:00:00PM', 'dd/mm/yyyy:hh:mi:ssam'), 39.99);

INSERT INTO NH\_BOOKING VALUES

(10, 10, 14, 8, 6, to\_date('20/10/2012:07:00:00PM', 'dd/mm/yyyy:hh:mi:ssam'), 49.99);

INSERT INTO NH\_BOOKING VALUES

(11, 1, 11, 5, 8, to\_date('17/07/2012:08:00:00PM', 'dd/mm/yyyy:hh:mi:ssam'), 59.99);

INSERT INTO NH\_BOOKING VALUES

(12, 4, 5, 11, 13, to\_date('17/07/2012:07:00:00PM', 'dd/mm/yyyy:hh:mi:ssam'), 43.3);

INSERT INTO NH\_BOOKING VALUES

(13, 7, 3, 10, 12, to\_date('17/07/2012:06:00:00PM', 'dd/mm/yyyy:hh:mi:ssam'), 117);

INSERT INTO NH\_BOOKING VALUES

(14, 8, 7, 12, 10, to\_date('18/09/2010:06:00:00PM', 'dd/mm/yyyy:hh:mi:ssam'), 65.4);

INSERT INTO NH\_BOOKING VALUES

(15, 2, 8, 14, 5, to\_date('12/12/2011:03:00:00PM', 'dd/mm/yyyy:hh:mi:ssam'), 23.6);

INSERT INTO NH\_BOOKING VALUES

(16, 7, 2, 4, 6, to\_date('14/09/2010:12:00:00PM', 'dd/mm/yyyy:hh:mi:ssam'), 32.3);

INSERT INTO NH\_BOOKING VALUES

(17, 4, 4, 2, 2, to\_date('04/08/2012:02:00:00PM', 'dd/mm/yyyy:hh:mi:ssam'), 56.7);

INSERT INTO NH\_BOOKING VALUES

(18, 12, 11, 8, 7, to\_date('16/07/2010:10:00:00PM', 'dd/mm/yyyy:hh:mi:ssam'), 55);

INSERT INTO NH\_BOOKING VALUES

(19, 13, 6, 15, 1, to\_date('16/07/2010:09:00:00PM', 'dd/mm/yyyy:hh:mi:ssam'), 42);

INSERT INTO NH\_BOOKING VALUES

(20, 1, 3, 1, 12, to\_date('20/07/2010:02:00:00PM', 'dd/mm/yyyy:hh:mi:ssam'), 49.99);

ALTER TABLE NH\_BOOKING

ADD CONSTRAINT Booking\_Customer\_FK FOREIGN KEY

(

CustomerID

)

REFERENCES NH\_CUSTOMER

(

CustomerID

)

;

ALTER TABLE NH\_BOOKING

ADD CONSTRAINT Booking\_Employee\_FK FOREIGN KEY

(

EmployeeID

)

REFERENCES NH\_EMPLOYEE

(

EmployeeID

)

;

ALTER TABLE NH\_BOOKING

ADD CONSTRAINT Booking\_Room\_FK FOREIGN KEY

(

RoomID

)

REFERENCES NH\_ROOM

(

RoomID

)

;

ALTER TABLE NH\_BOOKING

ADD CONSTRAINT Booking\_Treatment\_FK FOREIGN KEY

(

TreatmentID

)

REFERENCES NH\_TREATMENT

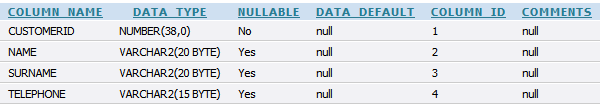
(

TreatmentID

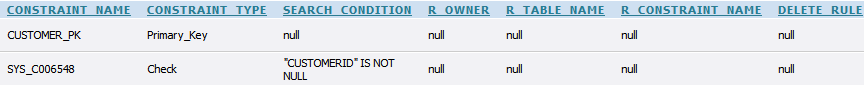
)

;

**NH\_CUSTOMER Table Structure**



**NH\_CUSTOMER Table Constraints**



**NH\_CUSTOMER Creation, Population and Constraint Script**

CREATE TABLE NH\_CUSTOMER

(

CustomerID INTEGER NOT NULL ,

Name VARCHAR2 (20) ,

Surname VARCHAR2 (20) ,

Telephone VARCHAR2 (15)

)

;

ALTER TABLE NH\_CUSTOMER

ADD CONSTRAINT Customer\_PK PRIMARY KEY ( CustomerID ) ;

/\* CustomerID INT, Name VARCHAR, Surname VARCHAR, Telephone VARCHAR \*/

INSERT INTO NH\_CUSTOMER VALUES

(1, 'aCustomerName', 'aCustomerSurname', '123456789');

INSERT INTO NH\_CUSTOMER VALUES (2, 'aCustomerName2', 'aCustomerSurname3', '123456789');

INSERT INTO NH\_CUSTOMER VALUES (3, 'Jack', 'New', '02089874563');

INSERT INTO NH\_CUSTOMER VALUES (4, 'Ryan', 'ken', '0207589989');

INSERT INTO NH\_CUSTOMER VALUES (5, ' Kai', 'roon', '02087414141');

INSERT INTO NH\_CUSTOMER VALUES (6, 'rick', 'nick', '02082020232');

INSERT INTO NH\_CUSTOMER VALUES (7, 'Adam', 'Ali', '0205255698');

INSERT INTO NH\_CUSTOMER VALUES (8, 'Ray', 'milton', '02085478745');

INSERT INTO NH\_CUSTOMER VALUES (9, 'milly', 'toof', '0208547457');

INSERT INTO NH\_CUSTOMER VALUES (10, 'Brown ', 'Joe', '02087414141');

INSERT INTO NH\_CUSTOMER VALUES (11, 'Lloyd', 'bank', '0208578785');

INSERT INTO NH\_CUSTOMER VALUES (12, 'Fred', 'Eil', '02085488878');

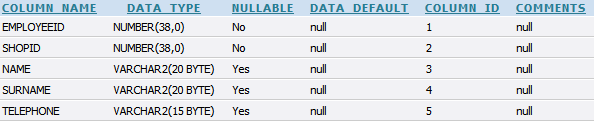
INSERT INTO NH\_CUSTOMER VALUES (13, 'Freddie', 'Elliot', '02085488878');

INSERT INTO NH\_CUSTOMER VALUES (14, 'Bill', 'bush', '02085555893');

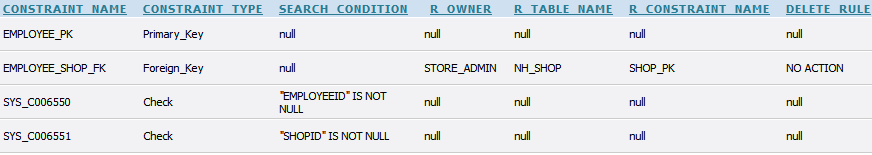
INSERT INTO NH\_CUSTOMER VALUES (15, 'red ', 'coe', '02085412569');

INSERT INTO NH\_CUSTOMER VALUES (16, 'Web', 'Ref', '02085411012');

**NH\_EMPLOYEE Table Structure**



**NH\_EMPLOYEE Table Constraints**



**NH\_EMPLOYEE Creation, Population and Constraint Script**

CREATE TABLE NH\_EMPLOYEE

(

EmployeeID INTEGER NOT NULL ,

ShopID INTEGER NOT NULL ,

Name VARCHAR2 (20) ,

Surname VARCHAR2 (20) ,

Telephone VARCHAR2 (15)

)

;

ALTER TABLE NH\_EMPLOYEE

ADD CONSTRAINT Employee\_PK PRIMARY KEY ( EmployeeID ) ;

/\* EmployeeID INT, ShopID INT, Name VARCHAR, Surname VARCHAR, Telephone VARCHAR \*/

INSERT INTO NH\_EMPLOYEE VALUES

(1, 1, 'anEmployeeName', 'anEmployeeSurname', '123456789');

INSERT INTO NH\_EMPLOYEE VALUES (2, 1, 'Manager', 'Badass', '123456789');

INSERT INTO NH\_EMPLOYEE VALUES (3,1, 'Employee', 'Tool', '123456789');

INSERT INTO NH\_EMPLOYEE VALUES (4, 2, 'Foreigner', 'nbd wants him', '123456789');

INSERT INTO NH\_EMPLOYEE VALUES (5, 2, 'Dwayne','Johnson', '02075122365');

INSERT INTO NH\_EMPLOYEE VALUES (6, 2, 'Rey','Riley', '012254589651');

INSERT INTO NH\_EMPLOYEE VALUES (7, 2, 'Kelly','Kelly', '01548596321');

INSERT INTO NH\_EMPLOYEE VALUES (8, 3, 'Zack','Rider', '020714788741');

INSERT INTO NH\_EMPLOYEE VALUES (9, 3, 'Alicia','Fox', '010136987456');

INSERT INTO NH\_EMPLOYEE VALUES (10, 4, 'Mohammed', 'Ali', '020845554874');

INSERT INTO NH\_EMPLOYEE VALUES (11, 5, 'Jay','Rich','012202012354');

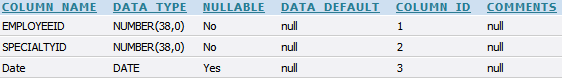
INSERT INTO NH\_EMPLOYEE VALUES (12, 5, 'Ben','Cole','020741113269');

INSERT INTO NH\_EMPLOYEE VALUES (13, 5, 'Steve', 'Law', '020741113269');

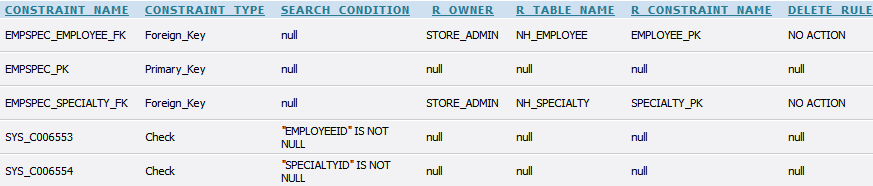
INSERT INTO NH\_EMPLOYEE VALUES (14, 6, 'Rock', 'Ross', '020741113269');

INSERT INTO NH\_EMPLOYEE VALUES (15, 7, 'Pat', 'Heart', '020741113269');

**NH\_EMPSPEC Table Structure**



**NH\_EMPSPEC Table Constraints**



NH\_EMPSPEC Creation, Population and Constraint Script

CREATE TABLE NH\_EMPSPEC

(

EmployeeID INTEGER NOT NULL ,

SpecialtyID INTEGER NOT NULL ,

"Date" DATE

)

;

ALTER TABLE NH\_EMPSPEC

ADD CONSTRAINT EmpSpec\_PK PRIMARY KEY ( EmployeeID, SpecialtyID ) ;

/\* EmployeeID INT, SpecialtyID INT, Date DATE \*/

INSERT INTO NH\_EMPSPEC VALUES

(1, 1, '12-DEC-2011');

INSERT INTO NH\_EMPSPEC VALUES (2, 1, '18-NOV-2011');

INSERT INTO NH\_EMPSPEC VALUES (5, 4, '04-FEB-2012');

INSERT INTO NH\_EMPSPEC VALUES (6, 10, '10-MAR-2012');

INSERT INTO NH\_EMPSPEC VALUES (9, 8, '26-APR-2012');

INSERT INTO NH\_EMPSPEC VALUES (6, 7, '27-OCT-2012');

INSERT INTO NH\_EMPSPEC VALUES (5, 3, '23-NOV-2011');

INSERT INTO NH\_EMPSPEC VALUES (7, 5, '12-OCT-2010');

INSERT INTO NH\_EMPSPEC VALUES (3, 4, '05-JUN-2012');

INSERT INTO NH\_EMPSPEC VALUES (4, 3, '12-DEC-2011');

INSERT INTO NH\_EMPSPEC VALUES (5, 5, '12-NOV-2012');

INSERT INTO NH\_EMPSPEC VALUES (6, 6, '12-MAR-2009');

INSERT INTO NH\_EMPSPEC VALUES (9, 9, '15-APR-2008');

INSERT INTO NH\_EMPSPEC VALUES (9, 3, '11-DEC-2010');

INSERT INTO NH\_EMPSPEC VALUES (5, 2, '20-DEC-2008');

INSERT INTO NH\_EMPSPEC VALUES (7, 7, '25-JAN-2011');

INSERT INTO NH\_EMPSPEC VALUES (6, 3, '27-MAR-2011');

INSERT INTO NH\_EMPSPEC VALUES (4, 8, '12-DEC-2007');

ALTER TABLE NH\_EMPSPEC

ADD CONSTRAINT EmpSpec\_Employee\_FK FOREIGN KEY

(

EmployeeID

)

REFERENCES NH\_EMPLOYEE

(

EmployeeID

)

;

ALTER TABLE NH\_EMPSPEC

ADD CONSTRAINT EmpSpec\_Specialty\_FK FOREIGN KEY

(

SpecialtyID

)

REFERENCES NH\_SPECIALTY

(

SpecialtyID

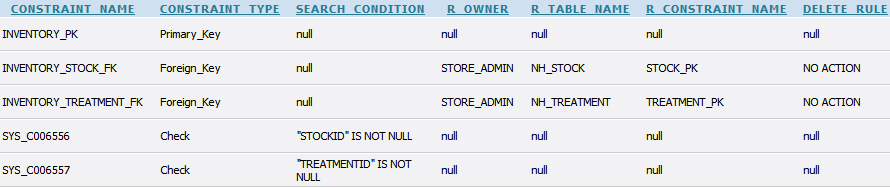
)

;

**NH\_INVENTORY Table Structure**



**NH\_INVENTORY Table Constraints**



**NH\_INVENTORY Table Creation, Population and Constraint Script**

CREATE TABLE NH\_INVENTORY

(

StockID INTEGER NOT NULL ,

TreatmentID INTEGER NOT NULL

)

;

ALTER TABLE NH\_INVENTORY

ADD CONSTRAINT Inventory\_PK PRIMARY KEY ( TreatmentID, StockID ) ;

/\* StockID INT, TreatmentID INT \*/

INSERT INTO NH\_INVENTORY VALUES

(1, 1);

INSERT INTO NH\_INVENTORY VALUES (2, 1);

INSERT INTO NH\_INVENTORY VALUES (3, 4);

INSERT INTO NH\_INVENTORY VALUES (4, 2);

INSERT INTO NH\_INVENTORY VALUES (5, 12);

INSERT INTO NH\_INVENTORY VALUES (6, 9);

INSERT INTO NH\_INVENTORY VALUES (7, 8);

INSERT INTO NH\_INVENTORY VALUES (8, 6);

INSERT INTO NH\_INVENTORY VALUES (9, 4);

INSERT INTO NH\_INVENTORY VALUES (10, 3);

INSERT INTO NH\_INVENTORY VALUES (11, 3);

INSERT INTO NH\_INVENTORY VALUES (12, 1);

ALTER TABLE NH\_INVENTORY

ADD CONSTRAINT Inventory\_Stock\_FK FOREIGN KEY

(

StockID

)

REFERENCES NH\_STOCK

(

StockID

)

;

ALTER TABLE NH\_INVENTORY

ADD CONSTRAINT Inventory\_Treatment\_FK FOREIGN KEY

(

TreatmentID

)

REFERENCES NH\_TREATMENT

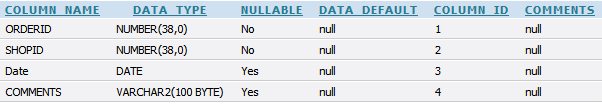
(

TreatmentID

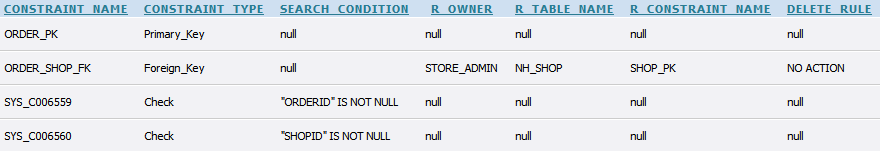
)

;

**NH\_ORDER Table Structure**



**NH\_ORDER Table Constraints**



**NH\_ORDER Table Creation, Population and Constraint Script**

CREATE TABLE NH\_ORDER

(

OrderID INTEGER NOT NULL ,

ShopID INTEGER NOT NULL ,

"Date" DATE ,

Comments VARCHAR2 (100)

)

;

ALTER TABLE NH\_ORDER

ADD CONSTRAINT Order\_PK PRIMARY KEY ( OrderID ) ;

/\* OrderID INT, ShopID INT, Date, DATE, Comments VARCHAR \*/

INSERT INTO NH\_ORDER VALUES

(1, 1, '13-JUN-2000', 'Delivery before July please');

INSERT INTO NH\_ORDER VALUES (2, 1, '22-FEB-2012', 'dont make a mess like last time');

INSERT INTO NH\_ORDER VALUES (3, 3, '22-DEC-2011', 'URGENT! BEFORE CHRISTMAS!');

INSERT INTO NH\_ORDER VALUES (4, 3, '23-DEC-2011', 'complementary to orderID 3');

INSERT INTO NH\_ORDER VALUES (5, 4, '23-FEB-2011', 'one package broken last time, be careful');

INSERT INTO NH\_ORDER VALUES (6, 6, '1-MAR-2011', 'before 10/3/2011 please');

INSERT INTO NH\_ORDER VALUES (7, 7, '24-MAR-2011', 'Its a national holiday in Greece tomorrow, we will have visitors');

INSERT INTO NH\_ORDER VALUES (8, 4, '04-APR-2011', 'Stock Refresh');

INSERT INTO NH\_ORDER VALUES (9, 6, '05-MAY-2011', 'Ordinary Scheduled');

INSERT INTO NH\_ORDER VALUES (10, 2, '04-JUN-2011', 'no comment');

INSERT INTO NH\_ORDER VALUES (11, 4, '07-JUL-2011', 'mid summer stock refreshment');

INSERT INTO NH\_ORDER VALUES (12, 7, '08-AUG-2011', 'just in case');

ALTER TABLE NH\_ORDER

ADD CONSTRAINT Order\_Shop\_FK FOREIGN KEY

(

ShopID

)

REFERENCES NH\_SHOP

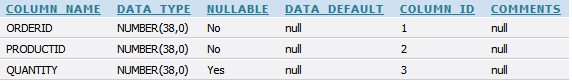
(

ShopID

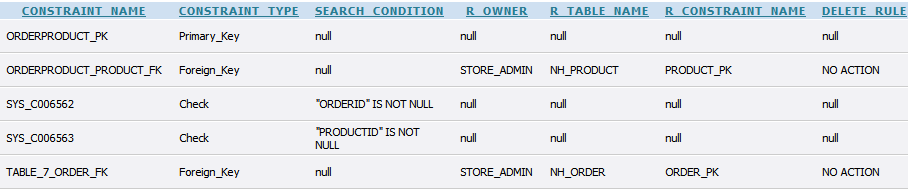
)

;

**NH\_ORDERPRODUCT Table Structure**



**NH\_ORDERPRODUCT Table Constraints**



**NH\_ORDERPRODUCT Table Creation, Population and Constraint Script**

CREATE TABLE NH\_ORDERPRODUCT

(

OrderID INTEGER NOT NULL ,

ProductID INTEGER NOT NULL ,

Quantity INTEGER

)

;

ALTER TABLE NH\_ORDERPRODUCT

ADD CONSTRAINT OrderProduct\_PK PRIMARY KEY ( OrderID, ProductID ) ;

/\* OrderID INT, ProductID INT, Quantity INT\*/

INSERT INTO NH\_ORDERPRODUCT VALUES

(1, 1, 3);

INSERT INTO NH\_ORDERPRODUCT VALUES (12, 15, 5);

INSERT INTO NH\_ORDERPRODUCT VALUES (11, 13, 6);

INSERT INTO NH\_ORDERPRODUCT VALUES (1, 12, 8);

INSERT INTO NH\_ORDERPRODUCT VALUES (4, 9, 4);

INSERT INTO NH\_ORDERPRODUCT VALUES (6, 5, 15);

INSERT INTO NH\_ORDERPRODUCT VALUES (3, 1, 8);

INSERT INTO NH\_ORDERPRODUCT VALUES (2, 1, 7);

INSERT INTO NH\_ORDERPRODUCT VALUES (6, 4, 23);

INSERT INTO NH\_ORDERPRODUCT VALUES (7, 7, 35);

INSERT INTO NH\_ORDERPRODUCT VALUES (2, 2, 8);

INSERT INTO NH\_ORDERPRODUCT VALUES (8, 4, 23);

INSERT INTO NH\_ORDERPRODUCT VALUES (10, 8, 15);

INSERT INTO NH\_ORDERPRODUCT VALUES (11, 12, 32);

INSERT INTO NH\_ORDERPRODUCT VALUES (9, 4, 1);

INSERT INTO NH\_ORDERPRODUCT VALUES (4, 5, 5);

INSERT INTO NH\_ORDERPRODUCT VALUES (5, 7, 80);

INSERT INTO NH\_ORDERPRODUCT VALUES (2, 4, 16);

ALTER TABLE NH\_ORDERPRODUCT

ADD CONSTRAINT OrderProduct\_Product\_FK FOREIGN KEY

(

ProductID

)

REFERENCES NH\_PRODUCT

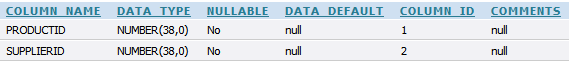
(

ProductID

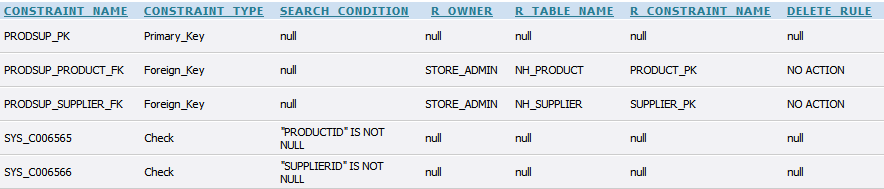
)

;

**NH\_PRODSUP Table Structure**



**NH\_PRODSUP Table Constraints**



**NH\_PRODSUP Table Creation, Population and Constraint Script**

CREATE TABLE NH\_PRODSUP

(

ProductID INTEGER NOT NULL ,

SupplierID INTEGER NOT NULL

)

;

ALTER TABLE NH\_PRODSUP

ADD CONSTRAINT ProdSup\_PK PRIMARY KEY ( ProductID, SupplierID ) ;

/\* ProductID INT, SupplierID INT \*/

INSERT INTO NH\_PRODSUP VALUES

(1, 1);

INSERT INTO NH\_PRODSUP VALUES (1, 2);

INSERT INTO NH\_PRODSUP VALUES (2, 2);

INSERT INTO NH\_PRODSUP VALUES (1, 5);

INSERT INTO NH\_PRODSUP VALUES (1, 6);

INSERT INTO NH\_PRODSUP VALUES (5, 1);

INSERT INTO NH\_PRODSUP VALUES (8, 1);

INSERT INTO NH\_PRODSUP VALUES (1, 7);

INSERT INTO NH\_PRODSUP VALUES (3, 1);

INSERT INTO NH\_PRODSUP VALUES (6, 8);

INSERT INTO NH\_PRODSUP VALUES (4, 3);

INSERT INTO NH\_PRODSUP VALUES (7, 1);

INSERT INTO NH\_PRODSUP VALUES (8, 8);

INSERT INTO NH\_PRODSUP VALUES (7, 6);

INSERT INTO NH\_PRODSUP VALUES (3, 4);

INSERT INTO NH\_PRODSUP VALUES (2, 6);

INSERT INTO NH\_PRODSUP VALUES (10, 11);

ALTER TABLE NH\_PRODSUP

ADD CONSTRAINT ProdSup\_Product\_FK FOREIGN KEY

(

ProductID

)

REFERENCES NH\_PRODUCT

(

ProductID

)

;

ALTER TABLE NH\_PRODSUP

ADD CONSTRAINT ProdSup\_Supplier\_FK FOREIGN KEY

(

SupplierID

)

REFERENCES NH\_SUPPLIER

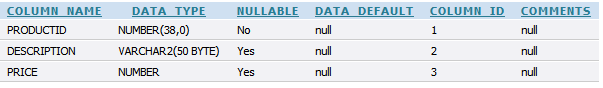
(

SupplierID

)

;

**NH\_PRODUCT Table Structure**



**NH\_PRODUCT Table Constraints**



**NH\_PRODUCT Table Creation, Population and Constraint Script**

CREATE TABLE NH\_PRODUCT

(

ProductID INTEGER NOT NULL ,

Description VARCHAR2 (50) ,

Price NUMBER

)

;

ALTER TABLE NH\_PRODUCT

ADD CONSTRAINT Product\_PK PRIMARY KEY ( ProductID ) ;

/\* ProductID INT, Desc VARCHAR, Price NUMBER (DECIMAL) \*/

INSERT INTO NH\_PRODUCT VALUES

(1, 'ahughes co', 15);

INSERT INTO NH\_PRODUCT VALUES (2, 'Amazon Hair', 13.2);

INSERT INTO NH\_PRODUCT VALUES (3, 'Latex Gloves', 2.7);

INSERT INTO NH\_PRODUCT VALUES (4, 'Premium Lotion', 22.75);

INSERT INTO NH\_PRODUCT VALUES (5, 'Ultra Premium Lambda Olive Oil', 49.99);

INSERT INTO NH\_PRODUCT VALUES (6, 'Syringe', 3.7);

INSERT INTO NH\_PRODUCT VALUES (7, 'My Lucky One', 11.7);

INSERT INTO NH\_PRODUCT VALUES (8, 'Massage Underwear', 0.75);

INSERT INTO NH\_PRODUCT VALUES (9, 'Massage Slippers', 1.99);

INSERT INTO NH\_PRODUCT VALUES (10, 'Hair Protector', 0.5);

INSERT INTO NH\_PRODUCT VALUES (11, 'Natural Hair Conditioner', 9.99);

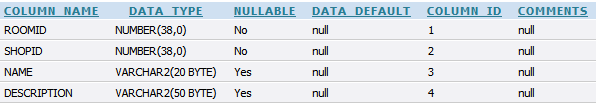
INSERT INTO NH\_PRODUCT VALUES (12, 'Natural Hair Shampoo', 4.99);

INSERT INTO NH\_PRODUCT VALUES (13, 'Relaxing Herbs', 3.99);

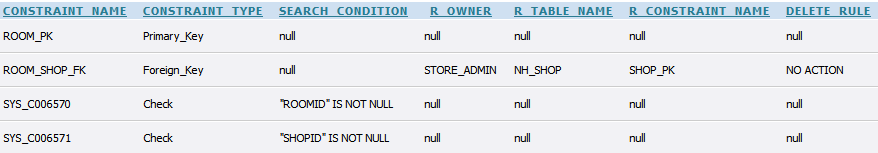
INSERT INTO NH\_PRODUCT VALUES (14, 'Natural Painkillers', 6.78);

INSERT INTO NH\_PRODUCT VALUES (15, 'Valerian Relaxing Pills', 3.98);

**NH\_ROOM Table Structure**



**NH\_ROOM Table Constraints**



**NH\_ROOM Table Creation, Population and Constraint Script**

CREATE TABLE NH\_ROOM

(

RoomID INTEGER NOT NULL ,

ShopID INTEGER NOT NULL ,

Name VARCHAR2 (20) ,

Description VARCHAR2 (50)

)

;

ALTER TABLE NH\_ROOM

ADD CONSTRAINT Room\_PK PRIMARY KEY ( RoomID ) ;

/\* RoomID INT, ShopID INT, Name VARCHAR, Desc VARCHAR \*/

INSERT INTO NH\_ROOM VALUES

(1, 1, 'G.01','Comfy');

INSERT INTO NH\_ROOM VALUES (2, 1, 'G.02','Richmans');

INSERT INTO NH\_ROOM VALUES (3, 1, 'F.01','Cozy');

INSERT INTO NH\_ROOM VALUES (4, 2, 'G.01','Jubilee');

INSERT INTO NH\_ROOM VALUES (5, 3, 'G.01','Estate');

INSERT INTO NH\_ROOM VALUES (6, 3, 'G.02','Mansion');

INSERT INTO NH\_ROOM VALUES (7, 5, 'E.12','Hand King');

INSERT INTO NH\_ROOM VALUES (8, 5, 'E.13','Foot King');

INSERT INTO NH\_ROOM VALUES (9, 5, 'F.12','21st century');

INSERT INTO NH\_ROOM VALUES (10, 6, 'G.01','Ergonomic');

INSERT INTO NH\_ROOM VALUES (11, 7, '117','Body Temple');

INSERT INTO NH\_ROOM VALUES (12, 7, '116','Physical Miracle');

INSERT INTO NH\_ROOM VALUES (13, 3, '1.01','Illumination');

ALTER TABLE NH\_ROOM

ADD CONSTRAINT Room\_Shop\_FK FOREIGN KEY

(

ShopID

)

REFERENCES NH\_SHOP

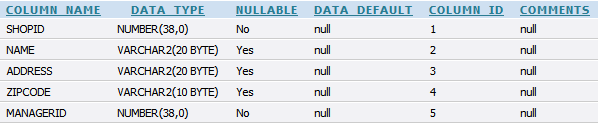
(

ShopID

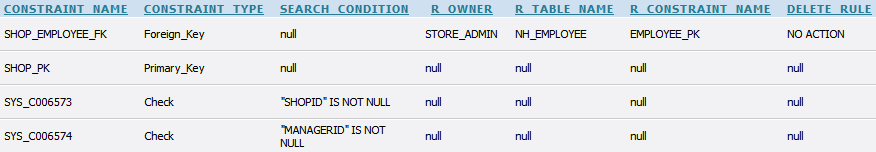
)

;

**NH\_SHOP Table Structure**



**NH\_SHOP Table Constraints**



**NH\_SHOP Table Creation, Population and Constraint Script**

CREATE TABLE NH\_SHOP

(

ShopID INTEGER NOT NULL ,

Name VARCHAR2 (20) ,

Address VARCHAR2 (20) ,

ZipCode VARCHAR2 (10) ,

ManagerID INTEGER NOT NULL

)

;

ALTER TABLE NH\_SHOP

ADD CONSTRAINT Shop\_PK PRIMARY KEY ( ShopID ) ;

/\* ShopID INT, Name VARCHAR, Address VARCHAR, Zip\_Code VARCHAR, ManagerID INT\*/

INSERT INTO NH\_SHOP VALUES

(1, 'aShopName', 'anAddress', 'aZip\_Code', 2);

INSERT INTO NH\_SHOP VALUES (2,'Bling','74 Addle Hill','E12 kiu', 4);

INSERT INTO NH\_SHOP VALUES (3,'Sexy SPA','19 Abchurch Yard','nw2 cfr', 8);

INSERT INTO NH\_SHOP VALUES (4,'The Hot Stop','95 Addle Street','s2 6za', 10);

INSERT INTO NH\_SHOP VALUES (5,'Relaxing','64Cutler Street','sw1 2lo', 11);

INSERT INTO NH\_SHOP VALUES (6,'Abbey','9 Cursitor Street','e13 98h', 14);

INSERT INTO NH\_SHOP VALUES (7,'Holy Bacon','45 Amen Corner ','n9 9np', 15);

ALTER TABLE NH\_SHOP

ADD CONSTRAINT Shop\_Employee\_FK FOREIGN KEY

(

ManagerID

)

REFERENCES NH\_EMPLOYEE

(

EmployeeID

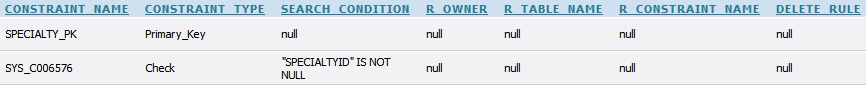
)

;

**NH\_SPECIALTY Table Structure**



**NH\_SPECIALTY Table Constraints**



**NH\_SPECIALTY Table Creation, Population and Constraint Script**

CREATE TABLE NH\_SPECIALTY

(

SpecialtyID INTEGER NOT NULL ,

Description VARCHAR2 (50)

)

;

ALTER TABLE NH\_SPECIALTY

ADD CONSTRAINT Specialty\_PK PRIMARY KEY ( SpecialtyID ) ;

/\*SpecialtyID INT, Description VARCHAR \*/

INSERT INTO NH\_SPECIALTY VALUES

(1, 'Head Massage');

INSERT INTO NH\_SPECIALTY VALUES (2, 'Neck Massage');

INSERT INTO NH\_SPECIALTY VALUES (3, 'Music and Aura Relaxation');

INSERT INTO NH\_SPECIALTY VALUES (4, 'Full Body Massage');

INSERT INTO NH\_SPECIALTY VALUES (5, 'Body Scrub');

INSERT INTO NH\_SPECIALTY VALUES (6, 'Face Scrub');

INSERT INTO NH\_SPECIALTY VALUES (7, 'Dry Hair Treatment');

INSERT INTO NH\_SPECIALTY VALUES (8, 'Curls and Head Scrub');

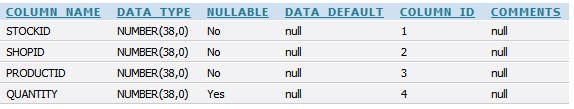
INSERT INTO NH\_SPECIALTY VALUES (9, 'Dry Skin Treatment');

INSERT INTO NH\_SPECIALTY VALUES (10, 'Psoriasis Treatment');

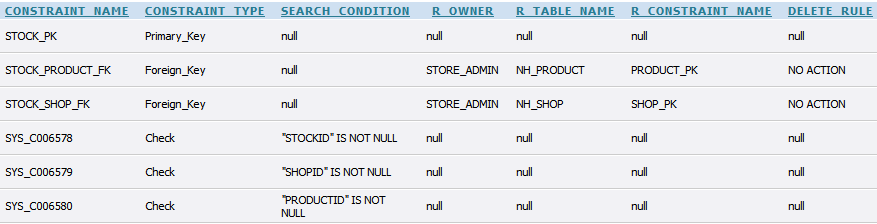
INSERT INTO NH\_SPECIALTY VALUES (11, 'Reiki Massage');

INSERT INTO NH\_SPECIALTY VALUES (12, 'Nail Polishing');

**NH\_STOCK Table Structure**



**NH\_STOCK Table Constraints**



**NH\_STOCK Table Creation, Population and Constraint Script**

CREATE TABLE NH\_STOCK

(

StockID INTEGER NOT NULL ,

ShopID INTEGER NOT NULL ,

ProductID INTEGER NOT NULL ,

Quantity INTEGER

)

;

ALTER TABLE NH\_STOCK

ADD CONSTRAINT Stock\_PK PRIMARY KEY ( StockID ) ;

/\* StockID INT, ShopID INT, ProductID INT, Quantity INT \*/

INSERT INTO NH\_STOCK VALUES

(1, 1, 1, 15);

INSERT INTO NH\_STOCK VALUES (2, 1, 12, 12);

INSERT INTO NH\_STOCK VALUES (3, 1, 14, 4);

INSERT INTO NH\_STOCK VALUES (4, 3, 3, 3);

INSERT INTO NH\_STOCK VALUES (5, 3, 10, 4);

INSERT INTO NH\_STOCK VALUES (6, 4, 9, 2);

INSERT INTO NH\_STOCK VALUES (7, 6, 15, 14);

INSERT INTO NH\_STOCK VALUES (8, 7, 1, 10);

INSERT INTO NH\_STOCK VALUES (9, 7, 2, 4);

INSERT INTO NH\_STOCK VALUES (10, 5, 12, 12);

INSERT INTO NH\_STOCK VALUES (11, 2, 14, 4);

INSERT INTO NH\_STOCK VALUES (12, 1, 3, 3);

INSERT INTO NH\_STOCK VALUES (13, 3, 4, 5);

INSERT INTO NH\_STOCK VALUES (14, 5, 13, 4);

INSERT INTO NH\_STOCK VALUES (15, 1, 4, 5);

ALTER TABLE NH\_STOCK

ADD CONSTRAINT Stock\_Product\_FK FOREIGN KEY

(

ProductID

)

REFERENCES NH\_PRODUCT

(

ProductID

)

;

ALTER TABLE NH\_STOCK

ADD CONSTRAINT Stock\_Shop\_FK FOREIGN KEY

(

ShopID

)

REFERENCES NH\_SHOP

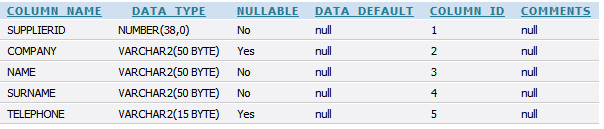
(

ShopID

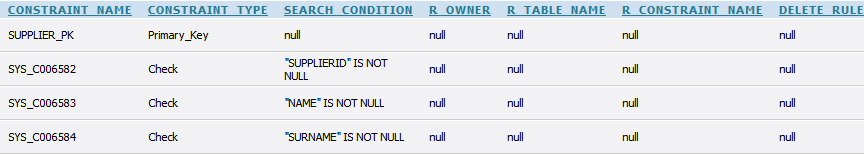
)

;

**NH\_SUPPLIER Table Structure**



**NH\_SUPPLIER Table Constraints**



**NH\_SUPPLIER Table Creation, Population and Constraint Script**

CREATE TABLE NH\_SUPPLIER

(

SupplierID INTEGER NOT NULL ,

Company VARCHAR2 (50) ,

Name VARCHAR2 (50) NOT NULL ,

Surname VARCHAR2 (50) NOT NULL ,

Telephone VARCHAR2 (15)

)

;

ALTER TABLE NH\_SUPPLIER

ADD CONSTRAINT Supplier\_PK PRIMARY KEY ( SupplierID ) ;

/\* SupplierID INT, Company VARCHAR, Name VARCHAR, Surname VARCHAR, Telephone VARCHAR \*/

INSERT INTO NH\_SUPPLIER VALUES

(1, 'HeadCare', 'Jack', 'Lee', '07908878547');

INSERT INTO NH\_SUPPLIER VALUES (2, 'HandLove', 'Red', 'Clif', '07505512336');

INSERT INTO NH\_SUPPLIER VALUES (3, 'FootLook', 'Kelly', 'Kent', '020874121458');

INSERT INTO NH\_SUPPLIER VALUES (4, 'BodyCare', 'Jeff', 'Kent', '07814777874');

INSERT INTO NH\_SUPPLIER VALUES (5, 'LookGood', 'Jimmy', 'Jay', '02085477777');

INSERT INTO NH\_SUPPLIER VALUES (6, 'FirstWax', 'Ian', 'Beat', '079022121456');

INSERT INTO NH\_SUPPLIER VALUES (7, 'Cleaning', 'Ken', 'Naz', '020836987451');

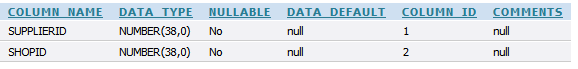
INSERT INTO NH\_SUPPLIER VALUES (8, 'Marks and Spencer', 'Melinda', 'Belius', '0759874521');

INSERT INTO NH\_SUPPLIER VALUES (9, 'Dont judge our name', 'George', 'Jack-the-Elfs', '07569874521');

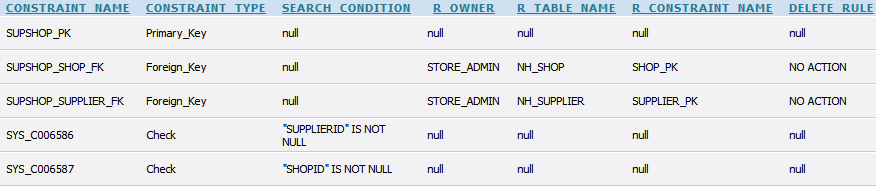
INSERT INTO NH\_SUPPLIER VALUES (10, 'Lousy Cleaning', 'George', 'Lazyman', '0756594521');

INSERT INTO NH\_SUPPLIER VALUES (11, 'Croco-Skin', 'Sting', 'Tooth', '07569874521');

**NH\_SUPSHOP Table Structure**

****

**NH\_SUPSHOP Table Constraints**

****

**NH\_SUPSHOP Table Creation, Population and Constraint Script**

CREATE TABLE NH\_SUPSHOP

(

SupplierID INTEGER NOT NULL ,

ShopID INTEGER NOT NULL

)

;

ALTER TABLE NH\_SUPSHOP

ADD CONSTRAINT SupShop\_PK PRIMARY KEY ( SupplierID, ShopID ) ;

/\* SupplierID INT, ShopID INT \*/

INSERT INTO NH\_SUPSHOP VALUES

(1, 1);

INSERT INTO NH\_SUPSHOP VALUES (11, 5);

INSERT INTO NH\_SUPSHOP VALUES (11, 1);

INSERT INTO NH\_SUPSHOP VALUES (1, 6);

INSERT INTO NH\_SUPSHOP VALUES (6, 2);

INSERT INTO NH\_SUPSHOP VALUES (4, 2);

INSERT INTO NH\_SUPSHOP VALUES (5, 4);

INSERT INTO NH\_SUPSHOP VALUES (5, 6);

INSERT INTO NH\_SUPSHOP VALUES (6, 3);

INSERT INTO NH\_SUPSHOP VALUES (1, 7);

INSERT INTO NH\_SUPSHOP VALUES (2, 1);

INSERT INTO NH\_SUPSHOP VALUES (6, 4);

INSERT INTO NH\_SUPSHOP VALUES (4, 3);

INSERT INTO NH\_SUPSHOP VALUES (10, 7);

INSERT INTO NH\_SUPSHOP VALUES (10, 5);

INSERT INTO NH\_SUPSHOP VALUES (5, 1);

INSERT INTO NH\_SUPSHOP VALUES (2, 2);

INSERT INTO NH\_SUPSHOP VALUES (9, 5);

ALTER TABLE NH\_SUPSHOP

ADD CONSTRAINT SupShop\_Shop\_FK FOREIGN KEY

(

ShopID

)

REFERENCES NH\_SHOP

(

ShopID

)

;

ALTER TABLE NH\_SUPSHOP

ADD CONSTRAINT SupShop\_Supplier\_FK FOREIGN KEY

(

SupplierID

)

REFERENCES NH\_SUPPLIER

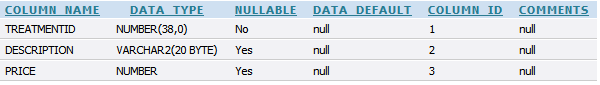
(

SupplierID

)

;

**NH\_TREATMENT Table Structure**

****

**NH\_TREATMENT Table Constraints**

****

**NH\_TREATMENT Table Creation, Population and Constraint Script**

CREATE TABLE NH\_TREATMENT

(

TreatmentID INTEGER NOT NULL ,

Description VARCHAR2 (20) ,

Price NUMBER

)

;

ALTER TABLE NH\_TREATMENT

ADD CONSTRAINT Treatment\_PK PRIMARY KEY ( TreatmentID ) ;

/\* TreatmentID INT, Desc VARCHAR, Price DECIMAL \*/

INSERT INTO NH\_TREATMENT VALUES

(1, 'Head Care', 15);

INSERT INTO NH\_TREATMENT VALUES (2, 'Foot Massage', 8.88);

INSERT INTO NH\_TREATMENT VALUES (3, 'Full Body Massage', 30);

INSERT INTO NH\_TREATMENT VALUES (4, 'Hair Loss Treatment', 100);

INSERT INTO NH\_TREATMENT VALUES (5, 'Leg Wax and Care', 20);

INSERT INTO NH\_TREATMENT VALUES (6, 'Spanish Massage', 55);

INSERT INTO NH\_TREATMENT VALUES (7, 'Face Massage', 15);

INSERT INTO NH\_TREATMENT VALUES (8, 'Face Scrub', 18);

INSERT INTO NH\_TREATMENT VALUES (9, 'Voice Therapy', 70);

INSERT INTO NH\_TREATMENT VALUES (10, 'Head Massage', 40);

INSERT INTO NH\_TREATMENT VALUES (11, 'Body Scrub', 60);

INSERT INTO NH\_TREATMENT VALUES (12, 'Stress Therapy', 28.99);

INSERT INTO NH\_TREATMENT VALUES (13, 'Full Body Massage', 50);

# 1.6 PL/SQL

## Stored function using a cursor.

/\* This function checks and returns the number of rows a particular query has. This query is the one who checks if we have date - room clashes and returns 0 if everything is alright (no rows fetched) and 1 if we need to check another time in the future.

\*\*\* We assume that booking are only closed exactly at an hour. e.g. 2 o’clock, NOT 2:15. \*\*\*

\*/

CREATE OR REPLACE FUNCTION

isRoomAvailable(

--input arguments: we only need room number and date (contains time)

v\_roomID IN INT,

v\_date IN NH\_BOOKING."Date"%TYPE)

--we return only whether we had rows or not.

RETURN NUMBER

IS

--output variable

v\_output NUMBER;

--creating the cursor, using the suitable query

--just selecting any integer type column, the result is not used

CURSOR queryCursor IS

SELECT RoomID

FROM NH\_BOOKING

WHERE RoomID = v\_roomID AND "Date" = v\_date;

BEGIN

--opening and iterating through cursor

OPEN queryCursor;

FETCH queryCursor INTO v\_output;

--if we have rows on resultset, means that the room is already booked.

IF queryCursor%FOUND THEN

v\_output := 1;

ELSE

v\_output := 0;

END IF;

--closing resources is a good habit

CLOSE queryCursor;

--returning result

RETURN v\_output;

EXCEPTION

WHEN OTHERS THEN

raise\_application\_error(-2001, 'Critical Error, call ahughes' || SQLERRM);

END;

## Triggers

/\* This trigger autoincrements a primary key value and makes insertions easier, for table NH\_BOOKING.

\*/

CREATE OR REPLACE SEQUENCE BOOKING\_SEQ;

CREATE OR REPLACE TRIGGER BOOKINGID\_AI

BEFORE INSERT ON "NH\_BOOKING"

FOR EACH ROW

BEGIN

SELECT "BOOKING\_SEQ".NEXTVAL INTO :NEW.BookingID FROM DUAL;

END;

/\* This trigger calculates and applies a discount to the booking that comply with the conditions below:\*

\* 10% for the 3rd visit\* 20% for the 6th visit\* 30% for the 9th visit\*/

CREATE OR REPLACE TRIGGER discountBEFORE INSERT ON "NH\_BOOKING"FOR EACH ROWDECLARE v\_instances INTEGER;BEGIN --by using mod we limit it to 9 instances SELECT MOD(COUNT(\*), 9) INTO v\_instances FROM NH\_BOOKING WHERE CustomerID = :NEW.CustomerID; --hacking around the mutating trigger error. --instead of updating after the table, --we check before doing a minus 1 IF (v\_instances = 2) THEN :NEW.Price := :NEW.Price - 0.1\*:NEW.Price; DBMS\_OUTPUT.PUT\_LINE('3'); ELSIF (v\_instances = 5) THEN :NEW.Price := :NEW.Price - 0.2\*:NEW.Price; DBMS\_OUTPUT.PUT\_LINE('6'); ELSIF (v\_instances = 8) THEN :NEW.Price := :NEW.Price - 0.3\*:NEW.Price; DBMS\_OUTPUT.PUT\_LINE('9'); END IF;END;

/\* This trigger prints a message every time that stock reaches minimum = 1 \*/CREATE OR REPLACE TRIGGER STOCKAFTER UPDATE ON "NH\_STOCK"FOR EACH ROWDECLARE v\_quantity INTEGER; v\_stockID INTEGER; v\_productID INTEGER; v\_shopID INTEGER;BEGIN v\_quantity := :NEW.Quantity; v\_stockID := :NEW.StockID; v\_productID := :NEW.ProductID; v\_shopID := :NEW.ShopID; --if stock under 2 items advise. IF (:NEW.Quantity < 2) THEN DBMS\_OUTPUT.PUT\_LINE('Warning, minimum amount of stock reached.'); DBMS\_OUTPUT.PUT\_LINE('StockID: ' || v\_stockID); DBMS\_OUTPUT.PUT\_LINE('ProductID: ' || v\_productID); DBMS\_OUTPUT.PUT\_LINE('ShopID: ' || v\_shopID); END IF;END;

## Stored Procedure

/\* This procedure makes-inserts a booking after it checks about the availability of the room. It makes use of the AutoIncrementing trigger and also of the check availability function

\*\*\* We assume that bookings are only closed exactly at an hour. e.g. 2 o’clock, NOT 2:15. \*\*\*

\*/CREATE OR REPLACE PROCEDUREnew\_booking( v\_roomID IN INT, v\_employeeID IN INT, v\_customerID IN INT, v\_treatmentID IN INT, v\_date IN NH\_BOOKING."Date"%TYPE, v\_price IN NUMBER)ASv\_errors INTEGER;BEGIN v\_errors := isRoomAvailable(v\_roomID, v\_date); IF (v\_errors =0) THEN --key is produced using AI trigger INSERT INTO NH\_BOOKING(RoomID, EmployeeID, CustomerID, TreatmentID, "Date", Price) VALUES (v\_roomID, v\_employeeID, v\_customerID, v\_treatmentID, v\_date, v\_price); DBMS\_OUTPUT.PUT\_LINE('Booking Completed!'); DBMS\_OUTPUT.PUT\_LINE('Customer must pay: ' || to\_char(v\_price)); COMMIT; ELSE DBMS\_OUTPUT.PUT\_LINE('This room is not available at that time. Try another one'); END IF; EXCEPTION WHEN OTHERS THEN raise\_application\_error(-20001, 'Critical Error, call ahughes ' || SQLERRM); END;

# 1.7 Showing the Queries

**4. Can you list all the staff names and their manager’s name, for a particular shop?**

SELECT E.Name AS Emp\_Name, E.Surname AS Emp\_Surname,

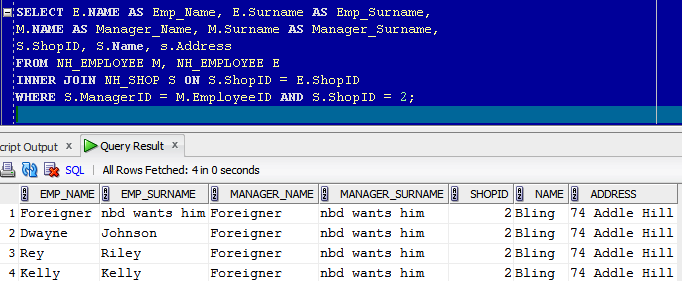
M.Name AS Manager\_Name, M.Surname AS Manager\_Surname,

S.ShopID, S.Address

FROM NH\_EMPLOYEE M, NH\_EMPLOYEE E

INNER JOIN NH\_SHOP S ON S.ShopID = E.ShopID

WHERE S.ManagerID = M.EmployeeID AND S.ShopID = 2;



**5. Can you list all the sessions a specific member of staff had on a specific day, which**

**customers and the room numbers?**

SELECT B.BookingID, B."Date", C.Name, C.Surname, R.RoomID,

R.Name AS Room\_Name

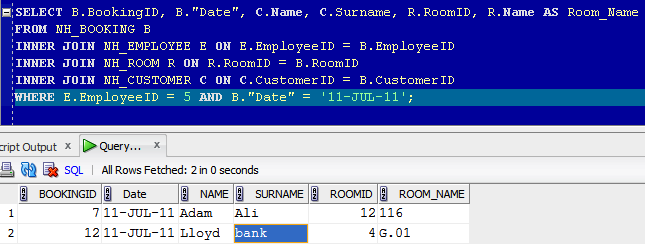
FROM NH\_BOOKING B

INNER JOIN NH\_EMPLOYEE E ON E.EmployeeID = B.EmployeeID

INNER JOIN NH\_ROOM R ON R.RoomID = B.RoomID

INNER JOIN NH\_CUSTOMER C ON C.CustomerID = B.CustomerID

WHERE E.EmployeeID = 5 AND B."Date" = '11-JUL-11';



**6. A customer could ask ‘Which member of staff did my 9-10 session today and can you tell me who their manager is because I want to complain?’**

SELECT B.BookingID, TO\_CHAR(B."Date",'hh24:mi:ss') AS DATETIME,

E.NAME AS Emp\_Name, E.Surname AS Emp\_Surname,

M.Name AS Mgr\_Name, M.Surname AS Mgr\_Surname

FROM NH\_EMPLOYEE M, NH\_BOOKING B

INNER JOIN NH\_CUSTOMER C ON C.CustomerID = B.CustomerID

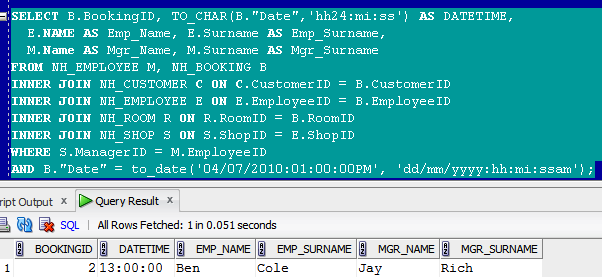
INNER JOIN NH\_EMPLOYEE E ON E.EmployeeID = B.EmployeeID

INNER JOIN NH\_ROOM R ON R.RoomID = B.RoomID

INNER JOIN NH\_SHOP S ON S.ShopID = E.ShopID

WHERE S.ManagerID = M.EmployeeID

AND B."Date" = to\_date('04/07/2010:01:00:00PM', 'dd/mm/yyyy:hh:mi:ssam');



# 1.8 Group Member Participation

Within our group we had people with some strengths and weaknesses who played a big part on the development of the database management system. We will try to present and report for some problems we had during this assignment.

Firstly let’s talk about the group members participation within the group.

Talking about **Mohammod Habibur Rahman (ID: u10080150)**, he is a very hard working person, he does a lot of research and he is very committed to his work and to his group team. He is one of them guys which will stay late to make sure that the DBMS is the best it could be within the team. This is a very strong attribute from Habib, however everyone has his own weakness, and for Habib his main weakness is that he has a disability which it hard for him some of the time to spell words. He’s dyslexic and this is one of his weaknesses; however this did not stop him from giving his most best 100% of the time while developing this system for the company. Another thing which I needed to develop on before I could really help out in the project was to expand my own key knowledge of database system and how to write simple PL/SQL. Once I did his I could start coding some of the database system for the company.

Apart from this one weakness which he has, his work which he has put in must be recognition as he has work as hard as he can to try and make this DBMS as best as possible as it can be. I cannot really say anything negative about this person as he is always the first to come in and the last to leave.

We both took turns in developing the database system as we both had to gain some new skills into learning PL/SQL. The way we went about this was to research a little about how to write PL/SQL before starting to develop the database management system for the company. Once we had learnt the PL/SQL we then had to expand our knowledge into database system in order to make the system function to the best of its ability.

Another aspect of the project which me and my group partner did was running the test on the database system to make sure that the system which was developed ran as smoothly as possible.

The second member of the group is **Alexandros Akrivopoulos-Hughes (ID: u1235441)**. The database coding was done by both of us. Both of us help each other to make sure that when we were developing the database system we used each other skills sets to achieve our main object.

My group member was the chief architect in designing the ER1 logical diagram, along with engineering it to ER2 relational diagram. He proposed sample diagrams commenting about possible problems that may arise from each of them and we both chose the best for the job.

My group member also did the most of the coding for creating the embedded business logic for the database. I had some skills in PL/SQL; however my group member had a better understanding of the big picture when it comes to PL/SQL. Therefore he was the one that designed the business logic functionality for the database but we both were charged with developing it.

These are the few things which we have done within the group. We were meeting up at least once a week to catch up and report for our progress and also make some decisions about the projects working flow. Within the group we had to understand each other’s strengths and weaknesses to make sure that each one of us was capable of delivering high quality work, so that the overall database solution was of high quality.

# 2.0 Evaluation

We were given a case study whereby we had to develop a database management system to help a company’s booking policy as well as stock management. The database system must be developed to help this company to minimize their errors in order to maximize their profits.

Once the ER1 logical diagram was created after brainstorming and merging proposed solutions from team members, we started testing it by creating real time scenarios and questions that the diagram should have the ability to answer.

After creating a proper logical model, we only had to import it and pass it to the right tools and the ER2 relational model was just a matter of time to construct, test and finalize. However, checking, modelling and implementing all the constraints takes time and energy.

After the ER2 relational model was made completely, it was really easy to make the database creation script, as well as some sample test data. We moved on with that and we had the ability of already answering the questions that have been asked to us, after constructing the right queries.

The other time consuming part was writing the PL/SQL for embedding the business logic into the database, because it was a bit new to every member of the team and we spent some time learning and experimenting with its advantages, disadvantages and capabilities.

In the end, business logic was also coded and shipped along with the database product and it really enhanced the product’s operation as it stopped double bookings, applied automatically discounts etc.

# 2.1 Recommendation

If we had more time we both think that we could have learnt much more of PL/SQL and we would have tried to develop an enhanced database system for the company. However, according to the time given and the workload of our everyday lives, the system was really complete and stable, as we really made use of our available time and optimized the development process by increasing our productivity, using the right tools for the job.

Overall we can both say that our group gained a lot of experience and also learn new things, concepts, ideas and methodologies during this assignment.

2.2 References:

[1] Thomas, C. and Carolyn, B. (2008), Database Systems A Practical Approach to Design, Implementation, and Management, 5th ed.

[2] Thomas, C. and Carolyn, B. (2010), Database system Database Systems a Practical Approach to Design, Implementation, and Management

[3] Andy, O. (2004), Database, a self teaching guide