



*the*  
UNIVERSITY  
*of*  
GREENWICH

# COMP1556 - DB Applications



Authors: Trevor Kiggundu  
University of Greenwich  
4/6/2019

# Table of Contents

1	Introduction.....	3
1.1	Who worked on what?.....	3
2	Assumptions.....	4
3	Conceptual Model.....	7
4	Physical Model.....	8
5.	The Implementation of Applications .....	10
5.1	The INSERT commands for each table .....	10
1.	Table 1 : STAFFPOSITION: 2 records.....	10
2.	Table 2 : EQUIPTYPE: 2 records .....	11
3.	Table 3 : CHIP: 100 records .....	11
4.	Table 4 : PATIENT: 100 records .....	12
5.	Table 5 : WARD: 20 records.....	12
6.	Table 6 : SECTOR: 20 records.....	13
7.	Table 7 : RESTPERIOD : 30 records .....	14
8.	Table 8 : EQUIPMENT : 30 records .....	14
9.	Table 9 : EQUIPMENTUSAGE: 40 records.....	15
10.	Table 10 : STAFF: 100 records .....	15
11.	Table 11 : STAFFCHIP: 100 records .....	16
12.	Table 12 : TREATMENT: 5 records .....	16
13.	Table 13 : STAFFTREATMENT: 100 records.....	17
6	SQL Coding Tasks.....	18
6.1	Task 1.....	18
6.2	Task 2.....	19
6.3	Task 3.....	21
6.4	Task 4.....	23
7	Self-evaluation .....	24
8	Self-Marking.....	25



# 1 INTRODUCTION

---

The Wired Hospital was a project handed to us by an already successful private hospital looking to modernize its way of conducting simple tasks. Outdated systems already in place were unable to perform tasks such as tracking patients, medical equipment and medical staff using chips/sensors; as well as data advanced collection. We were called in to create a well-designed database to collect and analyze the data received from the chips implemented within wristbands and equipment that was distributed throughout the building(s). We made sure to use our prior knowledge of hospital systems, as well as our knowledge of database creation to make the system. Shown below is our proposed design to help improve the efficiency of the Wired Hospital system.

## 1.1 WHO WORKED ON WHAT?

<b>Name of Student</b>	<b>Trevor Kiggundu</b>	<b>Rupesan Kalaimohan</b>
% input to assumptions	50	50
% input to conceptual diagram	50	50
% input to physical diagram	50	50
% input to DB creation and population	50	50
% input to List A tasks	100	-
% input to List B tasks	-	100

## 2 ASSUMPTIONS

---

1. Staff Position: The coursework specification given to us vaguely describes the people working at the wired hospital as ‘medical staff’. Only later in the specification does it specify that there are different types of medical staff such as ‘doctors, nurses, trainees, etc.’. Based on this, and our prior knowledge/experiences visiting hospitals, we came to the conclusion that there would probably other different types of staff. The specification also asks for our design to allow ‘new types of job, locations and equipment to be added’ should it be necessary. We implemented this into our system by creating an entity called ‘staffposition’, within which different staff positions available at the wired hospital could be clearly identified.

staffposition		
PKsp	<u>staffpositionID</u>	NUMBER
	staffposition	VARCHAR2(50 CHAR)

2. Patient Treatments: The specification given to us only mentions ‘patient records’ regarding the patient’s actual interactions in the wired hospital. It also does, however, mention that if a patient already exists in the system, then new patient records should not be created for that patient as it allows previous ‘medical services’ to be tracked. Based on this, and our prior knowledge/experiences visiting hospitals, we came to the conclusion that there would be multiple patients that would have undergone previous treatments and more importantly, undergone different types of treatments. This is also directly linked to the ‘Staff Position’ heading above, as different types of medical staff provide different treatments. We implemented this into our system by creating an entity called ‘stafftreatments’, within which different treatments provided at the wired hospital could be clearly identified.

stafftreatment		
PKst	<u>stafftreatmentID</u>	NUMBER
FK	<b>staffID</b>	NUMBER
FK	<b>treatmentID</b>	NUMBER

- Rest Period: The specification mentions that previous patient records are to be kept the same, as the system to create patient records ‘exists already’. It also specifies that patients can also be re-admitted to the wired hospital for whatever reason. Based on this, and our prior knowledge/experiences visiting hospitals, we came to the conclusion that there would be multiple patients that would have spent a certain period of time at the hospital, and either got discharged indefinitely, or returned for further/different treatment. We implemented this into our system by creating an entity called ‘restperiod’, within which different rest periods for each patient could be easily viewed. This would also apply to new patients as well.

restperiod		
PKRp	<u>restperiodID</u>	NUMBER
FK	<b>patientID</b>	NUMBER
FK	<b>chipID</b>	NUMBER
	stayperiod	TIMESTAMP
	dischargeperiod	TIMESTAMP

- Equipment Usage (Time): The specification does not specifically allude to any major difficulties regarding the equipment used in the wired hospital. However, it does mention one of its goals, which is to ‘improve efficiency’ regarding locating, scheduling replacements, and actually replacing hospital equipment , thus making a better hospital system. Based on this, we made an assumption that improving efficiency would also require improved organization, and decided to create an entity called ‘equipmentusage’ that includes an extra feature allowing for the usage time (start and end) to be logged. This was our favorite idea, as it stemmed from our experience using lab equipment on an ID-checked, sign in-sign out basis in our own King William labs.

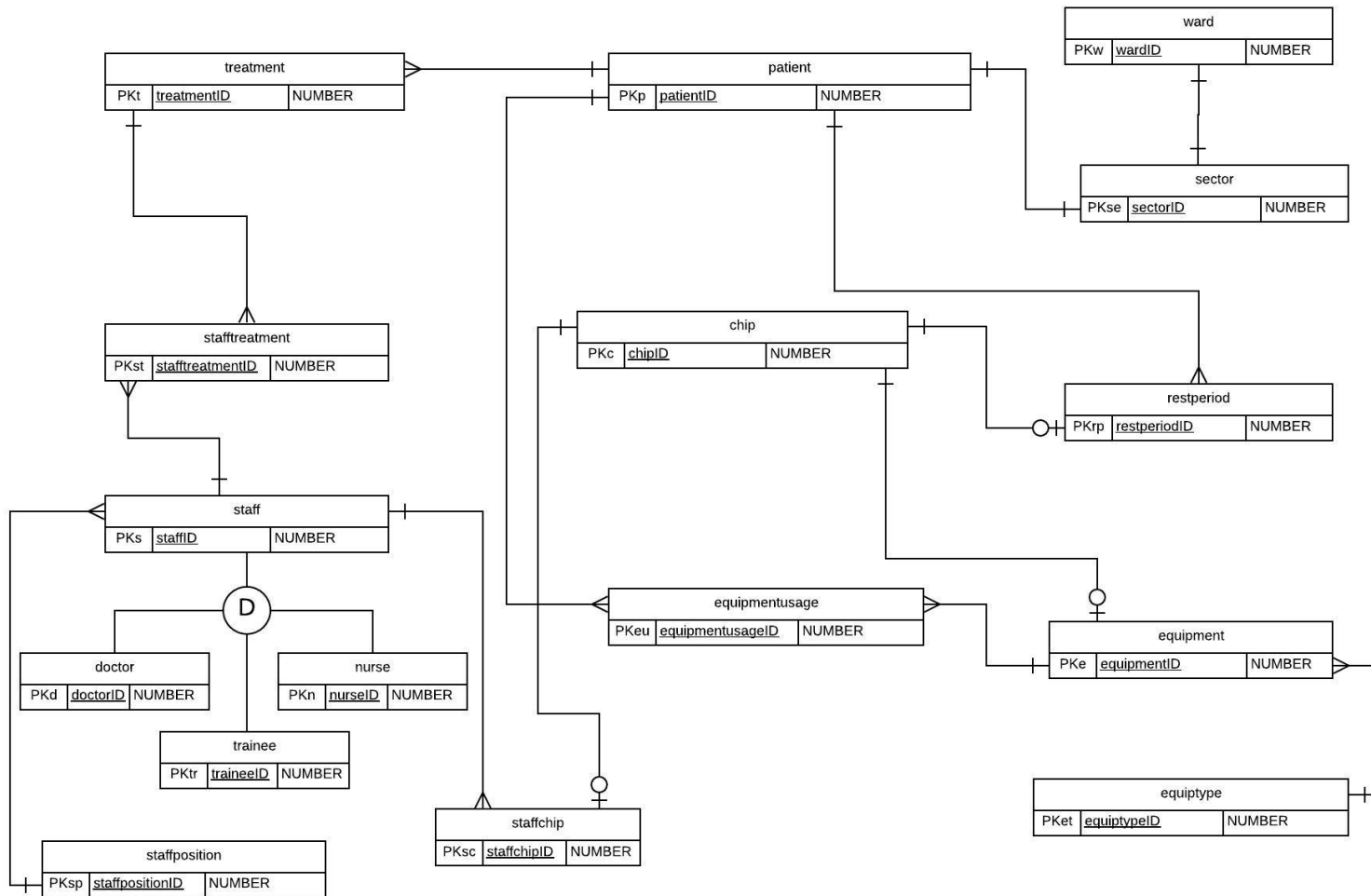
equipmentusage		
PKey	<u>equipmentusageID</u>	NUMBER
FK	<i>patientID</i>	NUMBER
FK	<i>equipmentID</i>	NUMBER
	startTime	TIMESTAMP
	endTime	TIMESTAMP

5. Equipment Usage (Type): The specification also lists in its goals for the improved Wired Hospital, that it would like to, as mentioned in assumption #1, have a design that allows ‘new types of job, locations and equipment to be added’. Based on this, we made an assumption that the hospital would have also plans to have the new system be implemented into new equipment that they possibly brought in in the near future, and not only in the equipment they currently have. We also assumed that they currently have much more equipment in the hospital than relayed to us in the specification. We implemented this into our system by creating an entity called ‘equiptype’, within which different types of equipment in the hospital could be easily identified. This would also apply to new types of equipment brought in as well.

equiptype		
PKey	<u>equiptypeID</u>	NUMBER
	equipmentType	VARCHAR2(50 CHAR)

### 3 CONCEPTUAL MODEL

## Conceptual



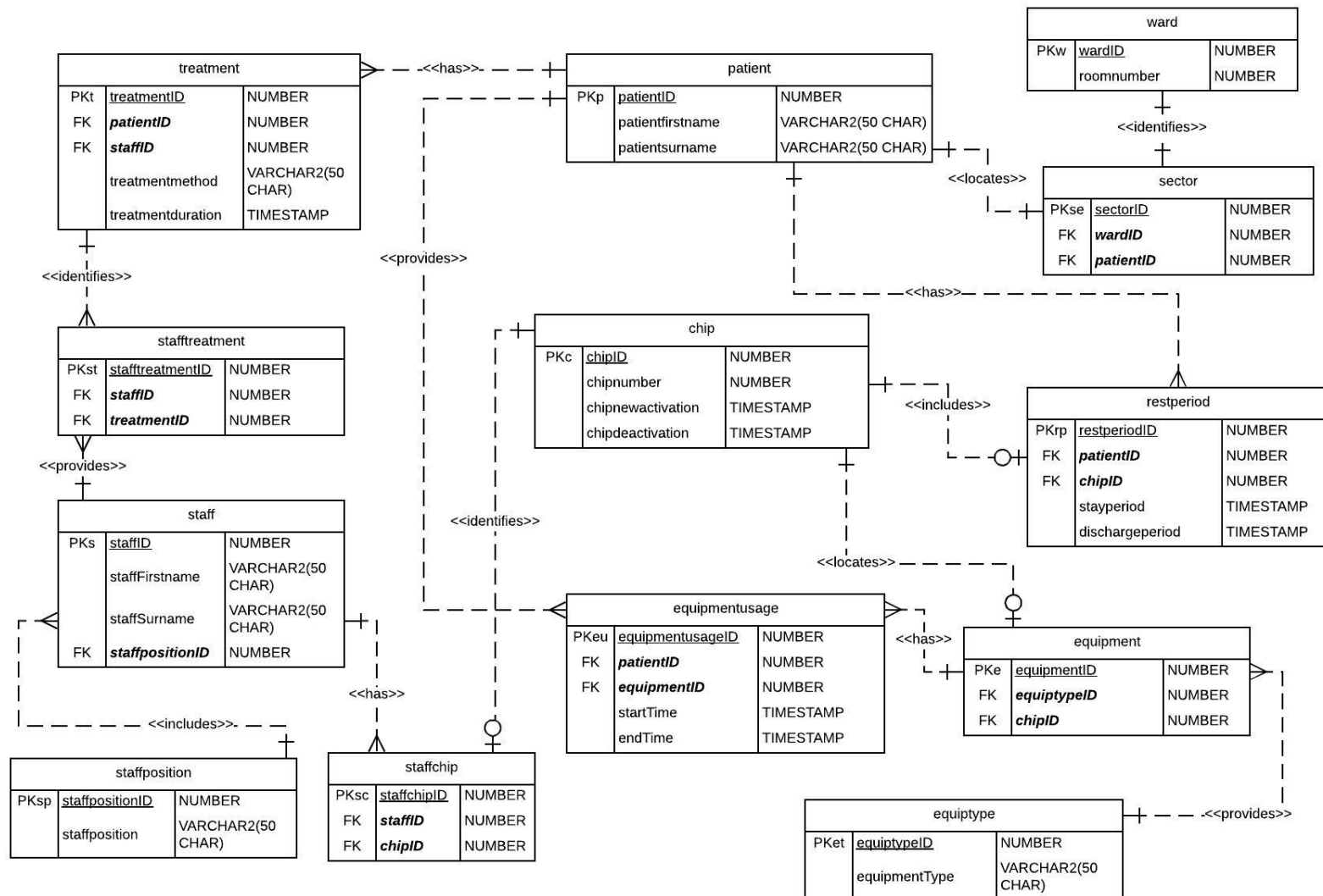


## 4 PHYSICAL MODEL

---

We implemented the six step process the same way that we were taught in the lectures. Starting with reading the scenario and finding the objects, we created a rough draft of the ERD and asked for feedback. This along with doing queries was the most strenuous part of completing the coursework, as we had to repeat multiple actions (such as redrawing) over and over again. Once we built a good foundation, we linked the entities using cardinality and added the PK's and FK's, making sure all necessary fields were present. We also made sure to keep our ERD's organized so that we would not get confused when we returned to work on them later. After 4 drafts, we were able to create one that we were satisfied with. Shown below is our completed physical ERD:

# Physical ERD

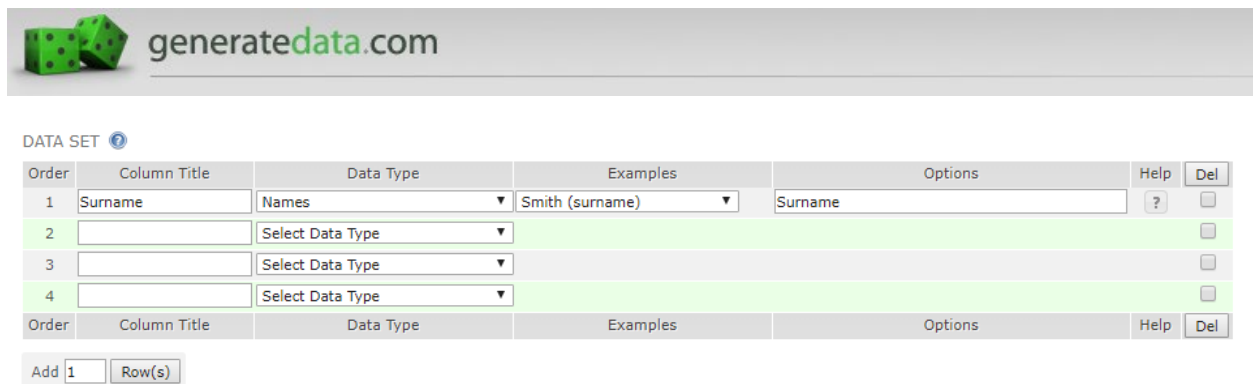


## 5. THE IMPLEMENTATION OF APPLICATIONS

---

### 5.1 THE INSERT COMMANDS FOR EACH TABLE

We created the tables using LucidChart, and used the export function to copy the code into Notepad ++ , where we edited and saved our code as .sql files. All the data that we used was generated using generatedata.com. Our population plan simply required inserting certain field names and data types (ex: firstname, surname), into the columns in generatedata.com to give us the correct parameters within which we could analyse data. Once we did that, the website did the rest and generated the data for us. Shown below are the insert commands and number of records added to each table.



The screenshot shows the generatedata.com website interface. At the top, there is a logo with two green dice and the text "generatedata.com". Below this, there is a section titled "DATA SET" with a help icon. The main part of the interface is a table configuration tool. It has a table with columns: Order, Column Title, Data Type, Examples, Options, Help, and Del. The table has 4 rows. Row 1 is filled with "Surname", "Names", "Smith (surname)", "Surname", "?", and a checkbox. Rows 2, 3, and 4 are empty, with "Select Data Type" in the Data Type column and checkboxes in the Del column. Below the table, there is a section with "Add" and "Row(s)" buttons.

Order	Column Title	Data Type	Examples	Options	Help	Del
1	Surname	Names	Smith (surname)	Surname	?	<input type="checkbox"/>
2		Select Data Type				<input type="checkbox"/>
3		Select Data Type				<input type="checkbox"/>
4		Select Data Type				<input type="checkbox"/>

Add  Row(s)

#### 1. Table 1 : STAFFPOSITION: 2 records

Code:

```
INSERT INTO staffposition  
(1,Doctor),  
  
INSERT INTO staffposition  
(2,Nurse),  
  
INSERT INTO staffposition  
(3,Trainee)  
  
;
```

Screenshots:

```
CREATE TABLE staffposition (
    staffpositionID NUMBER,
    staffposition VARCHAR2(50 CHAR),
    CONSTRAINT staffposition_pk PRIMARY KEY (staffpositionID)
);

COMMIT;
```

-- DATA FOR STAFFPOSITION TABLE

```
INSERT INTO staffposition (1,Doctor),
INSERT INTO staffposition (2,Nurse),
INSERT INTO staffposition (3,Trainee)
;
```

## 2. Table 2 : EQUIPTYPE: 2 records

Code:

```
INSERT INTO equiptype (1,Bed),
INSERT INTO equiptype
(2,Trolley),
INSERT INTO equiptype
(3,Monitor)
;
```

Screenshots:

```
CREATE TABLE equiptype (
    equiptypeID NUMBER,
    equipmentType VARCHAR2(50 CHAR),
    CONSTRAINT equiptype_pk PRIMARY KEY (equiptypeID)
);

COMMIT;
```

-- DATA FOR EQUIPTYPE TABLE

```
INSERT INTO equiptype (1,Bed),
INSERT INTO equiptype (2,Trolley),
INSERT INTO equiptype (3,Monitor)
;
```

## 3. Table 3 : CHIP: 100 records

Sample Code:

```
INSERT INTO chip (chipID, chipnumber, chipnewactivation, chipdeactivation)
VALUES (400,458,'12-Mar-19 05:25:53','24-Mar-19 05:56:50');
```

#### Screenshots:

```
CREATE TABLE chip (
  chipID NUMBER,
  chipnumber NUMBER,
  chipnewactivation TIMESTAMP,
  chipdeactivation TIMESTAMP,
  CONSTRAINT chip_pk PRIMARY KEY (chipID)
);

COMMIT;

-- DATA FOR CHIP TABLE

INSERT INTO chip (chipID,chipnumber,chipnewactivation,chipdeactivation) VALUES (400,458,'12-Mar-19 05:25:53','24-Mar-19 05:56:50');
INSERT INTO chip (chipID,chipnumber,chipnewactivation,chipdeactivation) VALUES (401,437,'17-Feb-19 09:45:25','09-Mar-19 21:42:43');
INSERT INTO chip (chipID,chipnumber,chipnewactivation,chipdeactivation) VALUES (402,500,'25-Feb-19 13:44:59','28-Jan-19 21:24:36');
INSERT INTO chip (chipID,chipnumber,chipnewactivation,chipdeactivation) VALUES (403,434,'17-Mar-19 06:22:34','04-Apr-19 11:36:04');
INSERT INTO chip (chipID,chipnumber,chipnewactivation,chipdeactivation) VALUES (404,460,'04-Apr-19 06:07:57','24-Feb-19 12:42:11');
INSERT INTO chip (chipID,chipnumber,chipnewactivation,chipdeactivation) VALUES (405,453,'30-Jan-19 21:09:00','29-Jan-19 18:06:08');
```

#### 4. Table 4 : PATIENT: 100 records

##### Sample Code:

```
INSERT INTO patient (patientID,patientfirstname,patientsurname) VALUES
(100,'Xerxes','Jacobson');
```

#### Screenshots:

```
CREATE TABLE patient (
  patientID NUMBER,
  patientfirstname VARCHAR2(50 CHAR),
  patientsurname VARCHAR2(50 CHAR),
  CONSTRAINT patient_pk PRIMARY KEY (patientID)
);

COMMIT;

-- DATA FOR PATIENT TABLE

INSERT INTO patient (patientID,patientfirstname,patientsurname) VALUES (100,'Xerxes','Jacobson');
INSERT INTO patient (patientID,patientfirstname,patientsurname) VALUES (101,'Cynthia','Madden');
INSERT INTO patient (patientID,patientfirstname,patientsurname) VALUES (102,'Justin','Dominguez');
INSERT INTO patient (patientID,patientfirstname,patientsurname) VALUES (103,'Suki','Underwood');
INSERT INTO patient (patientID,patientfirstname,patientsurname) VALUES (104,'Aurelia','Mays');
```

#### 5. Table 5 : WARD: 20 records

##### Sample Code:

```
INSERT INTO ward (wardID,roomnumber) VALUES (1,29);
```

Screenshot:

```
CREATE TABLE ward (  
  wardID NUMBER,  
  roomnumber NUMBER,  
  CONSTRAINT ward_pk PRIMARY KEY (wardID)  
);  
  
COMMIT;
```

-- DATA FOR WARD TABLE

```
INSERT INTO ward (wardID,roomnumber) VALUES (1,29) ;  
INSERT INTO ward (wardID,roomnumber) VALUES (2,49) ;  
INSERT INTO ward (wardID,roomnumber) VALUES (3,32) ;  
INSERT INTO ward (wardID,roomnumber) VALUES (4,24) ;  
INSERT INTO ward (wardID,roomnumber) VALUES (5,25) ;
```

6. Table 6 : SECTOR: 20 records

Sample Code:

```
INSERT INTO sector (sectorID,wardID,patientID) VALUES (1,16,126);
```

Screenshot:

```
CREATE TABLE sector (  
  sectorID NUMBER,  
  wardID NUMBER,  
  patientID NUMBER,  
  CONSTRAINT sector_pk PRIMARY KEY (sectorID)  
);  
  
COMMIT;
```

-- DATA FOR SECTOR TABLE

```
INSERT INTO sector (sectorID,wardID,patientID) VALUES (1,16,126);
INSERT INTO sector (sectorID,wardID,patientID) VALUES (2,18,118);
INSERT INTO sector (sectorID,wardID,patientID) VALUES (3,15,108);
INSERT INTO sector (sectorID,wardID,patientID) VALUES (4,18,111);
INSERT INTO sector (sectorID,wardID,patientID) VALUES (5,8,118);
```

## 7. Table 7 : RESTPERIOD : 30 records

Sample Code:

```
INSERT INTO restperiod
(restperiodID,patientID,chipID,stayperiod,dischargeperiod) VALUES
```

Screenshot:

```
CREATE TABLE restperiod (
  restperiodID NUMBER,
  patientID NUMBER,
  chipID NUMBER,
  stayperiod TIMESTAMP,
  dischargeperiod TIMESTAMP,
  CONSTRAINT restperiod_pk PRIMARY KEY (restperiodID)
);

COMMIT;
```

```
-- DATA FOR RESTPERIOD TABLE

INSERT INTO restperiod (restperiodID,patientID,chipID,stayperiod,dischargeperiod) VALUES (1,126,429,'31-Jan-19 14:02:41','28-Feb-19 10:38:56');
INSERT INTO restperiod (restperiodID,patientID,chipID,stayperiod,dischargeperiod) VALUES (2,128,416,'11-Apr-19 22:38:56','06-Feb-19 00:36:33');
INSERT INTO restperiod (restperiodID,patientID,chipID,stayperiod,dischargeperiod) VALUES (3,130,425,'19-Mar-19 10:25:18','12-Feb-19 13:24:48');
INSERT INTO restperiod (restperiodID,patientID,chipID,stayperiod,dischargeperiod) VALUES (4,124,402,'05-Apr-19 02:47:41','10-Feb-19 05:18:40');
INSERT INTO restperiod (restperiodID,patientID,chipID,stayperiod,dischargeperiod) VALUES (5,109,418,'01-Feb-19 21:09:03','25-Jan-19 04:42:30');
```

## 8. Table 8 : EQUIPMENT : 30 records

Sample Code:

```
INSERT INTO equipment (equipmentID,equiptypeID,chipID) VALUES (1,2,408);
```

Screenshot:

```
CREATE TABLE equipment (
  equipmentID NUMBER,
  equiptypeID NUMBER,
  chipID NUMBER,
  CONSTRAINT equipment_pk PRIMARY KEY (equipmentID)
);

COMMIT;
```

-- DATA FOR EQUIPMENT TABLE

```
INSERT INTO equipment (equipmentID,equiptypeID,chipID) VALUES (1,2,408);
INSERT INTO equipment (equipmentID,equiptypeID,chipID) VALUES (2,2,413);
INSERT INTO equipment (equipmentID,equiptypeID,chipID) VALUES (3,2,424);
INSERT INTO equipment (equipmentID,equiptypeID,chipID) VALUES (4,3,415);
INSERT INTO equipment (equipmentID,equiptypeID,chipID) VALUES (5,1,410);
```

## 9. Table 9 : EQUIPMENTUSAGE: 40 records

Sample Code:

```
INSERT INTO equipmentusage
(equipmentusageID,patientID,equipmentID,startTime,endTime) VALUES
(10,110,11,'19-Apr-19 08:53:51','09-Mar-19 07:06:34');
```

Screenshot:

```
CREATE TABLE equipmentusage (
  equipmentusageID NUMBER,
  patientID NUMBER,
  equipmentID NUMBER,
  startTime TIMESTAMP,
  endTime TIMESTAMP,
  CONSTRAINT equipmentusage_pk PRIMARY KEY (equipmentusageID)
);

COMMIT;
```

-- DATA FOR EQUIPMENTUSAGE TABLE

```
INSERT INTO equipmentusage (equipmentusageID,patientID,equipmentID,startTime,endTime) VALUES (10,110,11,'19-Apr-19 08:53:51','09-Mar-19 07:06:34');
INSERT INTO equipmentusage (equipmentusageID,patientID,equipmentID,startTime,endTime) VALUES (11,113,14,'21-Mar-19 00:36:06','05-Feb-19 14:00:44');
INSERT INTO equipmentusage (equipmentusageID,patientID,equipmentID,startTime,endTime) VALUES (12,104,29,'01-Mar-19 10:19:25','26-Feb-19 20:43:50');
INSERT INTO equipmentusage (equipmentusageID,patientID,equipmentID,startTime,endTime) VALUES (13,124,18,'18-Feb-19 07:33:19','22-Apr-19 02:23:53');
INSERT INTO equipmentusage (equipmentusageID,patientID,equipmentID,startTime,endTime) VALUES (14,125,21,'31-Jan-19 11:51:27','07-Apr-19 08:53:06');
INSERT INTO equipmentusage (equipmentusageID,patientID,equipmentID,startTime,endTime) VALUES (15,111,1,'27-Jan-19 18:42:46','02-Feb-19 20:48:34');
```

## 10. Table 10 : STAFF: 100 records

Sample Code:

```
INSERT INTO staff (staffID,staffFirstname,staffSurname,staffpositionID) VALUES
(200,'Jenna','Dixon',3);
```

Screenshot:

```
CREATE TABLE staff (
  staffID NUMBER,
  staffFirstname VARCHAR2(50 CHAR),
  staffSurname VARCHAR2(50 CHAR),
  staffpositionID NUMBER,
  CONSTRAINT staff_pk PRIMARY KEY (staffID)
);

COMMIT;
```



```
-- DATA FOR STAFF TABLE

INSERT INTO staff (staffID,staffFirstname,staffSurname,staffpositionID) VALUES (200,'Jenna','Dixon',3);
INSERT INTO staff (staffID,staffFirstname,staffSurname,staffpositionID) VALUES (201,'Winifred','Wilson',1);
INSERT INTO staff (staffID,staffFirstname,staffSurname,staffpositionID) VALUES (202,'Yoshio','Palmer',2);
INSERT INTO staff (staffID,staffFirstname,staffSurname,staffpositionID) VALUES (203,'Mira','Macias',2);
INSERT INTO staff (staffID,staffFirstname,staffSurname,staffpositionID) VALUES (204,'Ralph','Barnes',2);
```

## 11. Table 11 : STAFFCHIP: 100 records

Sample Code:

```
INSERT INTO staffchip (staffchipID,staffID,chipID) VALUES (300,235,454);
```

Screenshot:

```
CREATE TABLE staffchip (
  staffchipID NUMBER,
  staffID NUMBER,
  chipID NUMBER,
  CONSTRAINT staffchip_pk PRIMARY KEY (staffchipID)
);

COMMIT;
```

```
-- DATA FOR STAFFCHIP TABLE
```

```
INSERT INTO staffchip (staffchipID,staffID,chipID) VALUES (300,235,454);
INSERT INTO staffchip (staffchipID,staffID,chipID) VALUES (301,281,416);
INSERT INTO staffchip (staffchipID,staffID,chipID) VALUES (302,294,421);
INSERT INTO staffchip (staffchipID,staffID,chipID) VALUES (303,250,430);
INSERT INTO staffchip (staffchipID,staffID,chipID) VALUES (304,268,422);
```

## 12. Table 12 : TREATMENT: 5 records

Sample Code:

```
INSERT INTO treatment (1,109,210,MRE Scan,26-JAN-19 18.57.43.054000000),
```

Screenshot:

```
CREATE TABLE treatment (
  treatmentID NUMBER,
  patientID NUMBER,
  staffID NUMBER,
  treatmentmethod VARCHAR2(50 CHAR),
  treatmentduration TIMESTAMP,
  CONSTRAINT treatment_pk PRIMARY KEY (treatmentID)
);

COMMIT;
```

```
-- DATA FOR TREATMENT

INSERT INTO treatment (1,109,210,MRE Scan,26-JAN-19 18.57.43.054000000),
INSERT INTO treatment (2,125,235,Blood Test,30-JAN-19 18.57.50.146000000),
INSERT INTO treatment (3,130,240,MRE Scan,20-FEB-19 18.58.10.534000000),
INSERT INTO treatment (4,124,231,Injections,19-MAR-19 18.58.17.667000000),
INSERT INTO treatment (5,110,209,Blood Test,03-APR-19 18.58.23.231000000)
;
```

### 13. Table 13 : STAFFTREATMENT: 100 records

Sample Code:

```
INSERT INTO stafftreatment (stafftreatmentID,staffID,treatmentID) VALUES
(1,228,5);
```

Screenshot:

```
CREATE TABLE stafftreatment (
  stafftreatmentID NUMBER,
  staffID NUMBER,
  treatmentID NUMBER,
  CONSTRAINT stafftreatment_pk PRIMARY KEY (stafftreatmentID)
);

COMMIT;
```

```
-- DATA FOR STAFFTREATMENT

INSERT INTO stafftreatment (stafftreatmentID,staffID,treatmentID) VALUES (1,228,5);
INSERT INTO stafftreatment (stafftreatmentID,staffID,treatmentID) VALUES (2,269,3);
INSERT INTO stafftreatment (stafftreatmentID,staffID,treatmentID) VALUES (3,209,4);
INSERT INTO stafftreatment (stafftreatmentID,staffID,treatmentID) VALUES (4,235,4);
INSERT INTO stafftreatment (stafftreatmentID,staffID,treatmentID) VALUES (5,291,1);
```

## 6 SQL CODING TASKS

---

### 6.1TASK 1

**List : List A B**

## 6.2TASK 2

### List : List A2

**List the patients on a particular ward (entered at runtime) in family name/forename alphabetical order on a particular date (entered at runtime).**

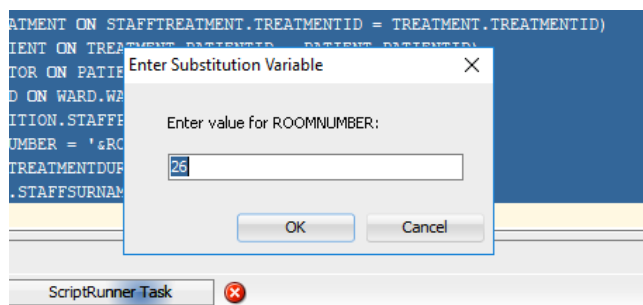
Code:

```
SELECT PATIENT.PATIENTSURNAME,
PATIENT.PATIENTFIRSTNAME,TREATMENT.TREATMENTDURATION, SECTOR.SECTORID

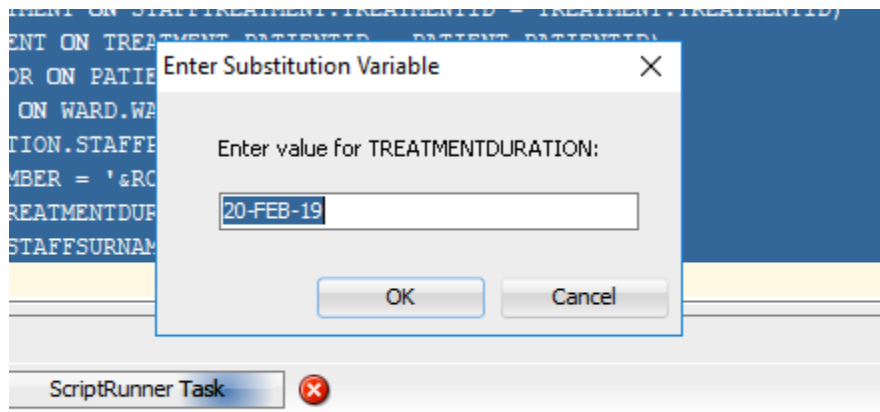
FROM ((((((STAFF
INNER JOIN STAFFPOSITION ON STAFF.STAFFPOSITIONID = STAFFPOSITION.STAFFPOSITIONID)
INNER JOIN STAFFTREATMENT ON STAFF.STAFFID = STAFFTREATMENT.STAFFID)
INNER JOIN TREATMENT ON STAFFTREATMENT.TREATMENTID = TREATMENT.TREATMENTID)
INNER JOIN PATIENT ON TREATMENT.PATIENTID = PATIENT.PATIENTID)
INNER JOIN SECTOR ON PATIENT.PATIENTID = SECTOR.PATIENTID)
INNER JOIN WARD ON WARD.WARDID = WARD.WARDID)
WHERE STAFFPOSITION.STAFFPOSITION = 'Nurse'
AND WARD.ROOMNUMBER = '&ROOMNUMBER'
AND TREATMENT.TREATMENTDURATION LIKE '%&TREATMENTDURATION%'
ORDER BY STAFF.STAFFSURNAME, STAFF.STAFFFIRSTNAME;
```

This query required me to list the patients in a particular ward by their family name in alphabetical order, as well as which date the event happened. The ward ID inputted was 26, and the date entered was 26, and the date inputted was 20<sup>th</sup> of February, 2019.

Ward Input:



Date Input:



Shown below are the results:

Script Output x

Task completed in 7.51 seconds

PATIENTSURNAME	PATIENTFIRSTNAME	TREATMENTDURATION	SI
Mathis	Griffin	20-FEB-19 18.58.10.534000000	
Mathis	Griffin	20-FEB-19 18.58.10.534000000	
Mathis	Griffin	20-FEB-19 18.58.10.534000000	
Mathis	Griffin	20-FEB-19 18.58.10.534000000	
Mathis	Griffin	20-FEB-19 18.58.10.534000000	
Mathis	Griffin	20-FEB-19 18.58.10.534000000	
Mathis	Griffin	20-FEB-19 18.58.10.534000000	

7 rows selected.

**List : List A3**

**List those patients treated by each nurse grouped by the family name of the nurse for those patients admitted in the last year:**

```
SELECT STAFF.STAFFSURNAME, PATIENT.PATIENTSURNAME,
PATIENT.PATIENTFIRSTNAME,TREATMENT.TREATMENTDURATION

FROM ((((((STAFF
INNER JOIN STAFFPOSITION ON STAFF.STAFFPOSITIONID =
STAFFPOSITION.STAFFPOSITIONID)
INNER JOIN STAFFTREATMENT ON STAFF.STAFFID =
STAFFTREATMENT.STAFFID)
INNER JOIN TREATMENT ON STAFFTREATMENT.TREATMENTID =
TREATMENT.TREATMENTID)
INNER JOIN PATIENT ON TREATMENT.PATIENTID = PATIENT.PATIENTID)
INNER JOIN SECTOR ON PATIENT.PATIENTID = SECTOR.PATIENTID)
INNER JOIN WARD ON WARD.WARDID = WARD.WARDID)
WHERE STAFFPOSITION.STAFFPOSITION = 'Nurse'

AND TREATMENT.TREATMENTDURATION LIKE
'%'&TREATMENTDURATION%'

ORDER BY STAFF.STAFFSURNAME, STAFF.STAFFFIRSTNAME;
```

This query required me to list the patients treated by each particular nurse over the last year. Our system only used data from, 2019, meaning that it would list all of the patients treated. The year entered was 19 (2019) and the results are shown below.

Script Output x			
Task completed in 2.592 seconds			
STAFFSURNAME	PATIENTSURNAME	PATIENTFIRSTNAME	TREATMENTDURATION
Walter	Mathis	Griffin	20-FEB-19 18.58.10.534000000
Script Output x			
Task completed in 2.592 seconds			
Wise	Mathis	Griffin	20-FEB-19 18.58.10.534000000
Wise	Mathis	Griffin	20-FEB-19 18.58.10.534000000
STAFFSURNAME	PATIENTSURNAME	PATIENTFIRSTNAME	TREATMENTDURATION
Wise	Mathis	Griffin	20-FEB-19 18.58.10.534000000
Wise	Mathis	Griffin	20-FEB-19 18.58.10.534000000
Wise	Mathis	Griffin	20-FEB-19 18.58.10.534000000

Oracle SQL Developer: obiwan

File Edit View Navigate Run Source Team Tools Window Help

StartPage obiwan 2.69799995 seconds

Worksheet Query Builder

```

SELECT STAFF.STAFFSURNAME, PATIENT.PATIENTSURNAME, PATIENT.PATIENTFIRSTNAME, TREATMENT.TREATMENTDURATION
FROM (((((STAFF
INNER JOIN STAFFPOSITION ON STAFF.STAFFPOSITIONID = STAFFPOSITION.STAFFPOSITIONID)
INNER JOIN STAFFTREATMENT ON STAFF.STAFFID = STAFFTREATMENT.STAFFID)
INNER JOIN TREATMENT ON STAFFTREATMENT.TREATMENTID = TREATMENT.TREATMENTID)
INNER JOIN PATIENT ON TREATMENT.PATIENTID = PATIENT.PATIENTID)
INNER JOIN SECTOR ON PATIENT.PATIENTID = SECTOR.PATIENTID)
INNER JOIN WARD ON WARD.WARDID = PATIENT.PATIENTID)
WHERE STAFFPOSITION.STAFFPOSITION = 'Nurse'

AND TREATMENT.TREATMENTDURATION LIKE '%TREATMENTDURATION%'
ORDER BY STAFF.STAFFSURNAME, STAFF.STAFFFIRSTNAME;

```

Script Output

Task completed in 2.698 seconds

Wise	Mathis	Griffin	20-FEB-19 18.58.10.534000000
Wise	Mathis	Griffin	20-FEB-19 18.58.10.534000000
STAFFSURNAME	PATIENTSURNAME	PATIENTFIRSTNAME	TREATMENTDURATION
Wise	Mathis	Griffin	20-FEB-19 18.58.10.534000000
Wise	Mathis	Griffin	20-FEB-19 18.58.10.534000000
Wise	Mathis	Griffin	20-FEB-19 18.58.10.534000000

300 rows selected.

## 6.4TASK 4

### List : List 4A

**List those beds that each doctor has not visited ordered by the ward in which the bed is located.**

This query required me to list the beds that the doctor had not visited by each ward.

Unfortunately, I was not able to complete this task before the deadline. I believe the errors occurred as a result of a misinterpretation of an earlier version on our ERD. However, shown below is the last version of code I planned to implement into Oracle.

```
SELECT staffFirstname, staffLastname, se.sector
FROM STAFF s, SECTOR se, EQUIPMENT e

INNER JOIN STAFFPOSITION sp
      ON s.STAFFID = sp.STAFFID
INNER JOIN WARD w
      ON se.SECTORID = w.SECTORID
INNER JOIN CHIP c
      ON se.CHIP = c.CHIP
WHERE EQUIPMENT = 'Bed'
ORDER BY staffFirstname, staffLastname
```



## 7 SELF-EVALUATION

---

The Database Applications Technologies was one of my most enjoyable modules since I joined the university last year. The fact that we worked in pairs gave me a happy medium between working alone and working in overly large groups that have caused stress in other/previous modules. Factors that went well were my ability to work well with my partner. I have worked with him before and we live in close proximity, so we were able to meet up when necessary outside of the tutorial times. We were able to set goals and complete our allocated tasks while still balancing other coursework activities, which was very satisfying to me. Things that did not go well were making mistakes regarding our Oracle programming, and even simple syntax errors in text files kept having us restart the process over and over just to find out that we were making one small mistake at a time. It was frustrating at times. However, we were able to pull through and submit our best work, so I am glad. As much as we were able to complete our diagrams and most of our queries, there were, as there is with anything, areas for improvement. We found out too late that we could have designed our diagrams a little bit better to allow us to make completing the queries a little easier. Making the ward and sector entities a one to one relationship would have resulted in us (my partner) being able to complete the queries much more efficiently than we did, saving us some time and stress. If given more time, we would have been able to create a much better system for the wired hospital. In this module, I learned how to create databases using SQL and Oracle, coding and queries, as well as perfecting ERD models. The LYW4 exercises were helpful as well, though I did not finish all of them. I do hope that this form of learning grows in popularity in the near future, and I am happy to say that I enjoyed the Database Applications Technologies module.

## 8 SELF-MARKING

---

	Out Of	Awarded
<b>Assumptions</b>	5	4

<b>Conceptual Model</b>	Out Of	Awarded
Identifying necessary main entities	5	
Identifying valid PKs	2	
Identifying sub-classes and their type	8	
Identifying relationships	5	
Cardinality of relationships	5	
Participation constraints of relationships	5	
<b>Sub-total</b>	30	24

<b>Physical Model</b>	Out Of	Awarded
Appropriate list of attributes	4	
Sub-classes and M-Ms resolved	8	
The correct placement of FKs	4	
Proof normalisation was carried out	4	
<b>Sub-total</b>	20	15

<b>Database Implementation</b>	Out Of	Awarded
Population plan and DB creation	4	

Population implementation

4

**Sub-total**

8

6

**Coursework Queries**

**Out of**

**List A**

**List B**

Query 1 - Joins

6

Query 2 - ORDER BY and Runtime input

8

Query 3 - GROUP BY ... HAVING

8

Query 4 - Sub-queries

10

**Sub-total**

32

19

0

**Out Of**

**Awarded**

**Evaluation**

5

4

**Out Of**

**Awarded**

**Grand Total**

100

72