1. Table Structures

SITE(*Site_Number*, Name, Manager_ID)
UNIT(*Unit_ID*, Unit_Number, Floor_Area, Zone, Site_Number)
EMPLOYEE(Employee_Number, Name, Job_Title, Home_Address, Telephone_Number)
MANAGER(*Manager_Number*, Employee_Number)
MENTORS(*Mentee_ID*, Mentor_ID)
SECRETARY(*Secretary_Number*, Employee_Number, Start_Date, Manager_Number)
JOBUNIT(*Ju_Number*, Unit_ID, Job_Number)
JOB(*Job_Number*, Description, Logdate)
EMERGENCY_JOBS(*E_Number*, Job_Number, E_Date_and_Time)
ROUTINE_JOBS(*R_Number*, Job_Number)

2. Creating Tables

```
SQL> CREATE TABLE EMPLOYEE (
        EMPLOYEE_NUMBER NUMBER NOT NULL PRIMARY KEY,
        NAME CHAR (20) NOT NULL,
        JOB_TITLE CHAR (20) NOT NULL,
        HOME_ADDRESS CHAR (50),
        TELEPHONE_NUMBER NUMBER (11));
Table created.
SQL> CREATE TABLE MANAGER (
        MANAGER_NUMBER NUMBER NOT NULL PRIMARY KEY,
        EMPLOYEE_NUMBER NUMBER NOT NULL,
        CONSTRAINT FK_EMPMAN FOREIGN KEY (EMPLOYEE_NUMBER)
        REFERENCES EMPLOYEE(EMPLOYEE_NUMBER));
Table created.
SQL> CREATE TABLE MENTOR (
        MENTEE_ID NUMBER NOT NULL PRIMARY KEY,
        MENTOR_ID NUMBER NOT NULL,
        CONSTRAINT FK_MENTEEMANAGER FOREIGN KEY (MENTEE_ID)
        REFERENCES MANAGER(MANAGER_NUMBER),
        CONSTRAINT FK_MENTORMANAGER FOREIGN KEY (MENTOR_ID)
        REFERENCES MANAGER(MANAGER_NUMBER));
Table created.
SQL> CREATE TABLE SECRETARY (
        SECRETARY_NUMBER NUMBER NOT NULL PRIMARY KEY,
        EMPLOYEE_NUMBER NUMBER NOT NULL,
        START_DATE DATE,
        MANAGER_NUMBER NUMBER NOT NULL,
        CONSTRAINT FK_EMPSEC FOREIGN KEY (EMPLOYEE_NUMBER)
        REFERENCES EMPLOYEE(EMPLOYEE_NUMBER),
        CONSTRAINT FK_SECMAN FOREIGN KEY (MANAGER_NUMBER)
        REFERENCES MANAGER(MANAGER_NUMBER));
Table created.
SQL> CREATE TABLE JOB (
        JOB_NUMBER NUMBER NOT NULL PRIMARY KEY,
        DESCRIPTION CHAR (90),
        LOGDATE DATE);
Table created.
SQL> CREATE TABLE EMERGENCY_JOBS (
        E_NUMBER NUMBER NOT NULL PRIMARY KEY,
```

```
        JOB_NUMBER NUMBER NOT NULL,
        E_DATE_AND_TIME DATE,
        CONSTRAINT FK_EJOB FOREIGN KEY (JOB_NUMBER)
        REFERENCES JOB(JOB_NUMBER));
Table created.
SQL> CREATE TABLE ROUTINE_JOBS (
        R_NUMBER NUMBER NOT NULL PRIMARY KEY,
        JOB_NUMBER NUMBER NOT NULL,
        CONSTRAINT FK_RJOB FOREIGN KEY (JOB_NUMBER)
        REFERENCES JOB(JOB_NUMBER));
Table created.
SQL> CREATE TABLE SITE (
        SITE_NUMBER NUMBER NOT NULL PRIMARY KEY,
        NAME CHAR (20) NOT NULL ,
        MANAGER_NUMBER NUMBER NOT NULL,
        CONSTRAINT FK_SITEMAN FOREIGN KEY (MANAGER_NUMBER)
        REFERENCES MANAGER(MANAGER_NUMBER));
Table created.
SQL> CREATE TABLE UNIT (
        UNIT_ID CHAR (3) NOT NULL PRIMARY KEY,
        UNIT_NUMBER NUMBER,
        FLOOR_AREA NUMBER,
        ZONE CHAR (20),
        SITE_NUMBER NUMBER NOT NULL,
        CONSTRAINT FK_UNITSITE FOREIGN KEY (SITE_NUMBER)
        REFERENCES SITE(SITE_NUMBER));
Table created.
SQL> CREATE TABLE JOBUNIT (
        JU_NUMBER NUMBER NOT NULL PRIMARY KEY,
        UNIT_ID CHAR (3) NOT NULL,
        JOB_NUMBER NUMBER NOT NULL,
        CONSTRAINT FK_JUUNIT FOREIGN KEY (UNIT_ID)
        REFERENCES UNIT(UNIT_ID),
        CONSTRAINT FK_JUJOB FOREIGN KEY (JOB_NUMBER)
        REFERENCES JOB(JOB_NUMBER));
Table created.
```

3. Populating Tables

```
SQL> INSERT INTO EMPLOYEE VALUES(1,'Charlie McKerracher','Manager','62 Park Avenue', 07854756981);
1 row created.
SQL> INSERT INTO EMPLOYEE VALUES(2,'Bridie Copse','Secretary','12 High Street',07700900221);
1 row created.
SQL> INSERT INTO EMPLOYEE VALUES(3,'Colm Meaney', 'Manager','24 Jamaica Street',07700900222);
1 row created.
SQL> INSERT INTO EMPLOYEE VALUES(4,'Ian McElhinney','Secretary','17 Patrick Way',07700900223);
1 row created.
SQL> INSERT INTO EMPLOYEE VALUES(5,'Tommy Tiernan','Manager','42 Towel Park',07700900224);
1 row created.
SQL> INSERT INTO EMPLOYEE VALUES(6,'Jack Sinclair','Secretary','221B Baker Street',07700900225);
1 row created.
SQL> INSERT INTO EMPLOYEE VALUES(7,'Ted Crilly','Manager','16 Pump Street',07700900226);
```

```
1 row created.
SQL> INSERT INTO EMPLOYEE VALUES(8,'Jack Hackett','Secretary','2 Magazine Street',07700900227);
1 row created.
SQL> INSERT INTO EMPLOYEE VALUES(9,'Hugh Jackman','Secretary','19 Artillery Street',07700900228);
1 row created.
SQL> INSERT INTO EMPLOYEE VALUES(10,'Wade Wilson','Secretary','7 Ballista Row',07700900229);
1 row created.
SQL> INSERT INTO MANAGER VALUES(1,1);
1 row created.
SQL> INSERT INTO MANAGER VALUES(2,3);
1 row created.
SQL> INSERT INTO MANAGER VALUES(3,5);
1 row created.
SQL> INSERT INTO MANAGER VALUES(4,7);
1 row created.
SQL> INSERT INTO MENTOR VALUES(1,2);
1 row created.
SQL> INSERT INTO MENTOR VALUES(3,4);
1 row created.
SQL> INSERT INTO SECRETARY VALUES(1,2,TO_DATE('12-JUN-2020','DD-MON-YYYY'),4);
1 row created.
SQL> INSERT INTO SECRETARY VALUES(2,4,TO_DATE('01-SEP-1975','DD-MON-YYYY'),3);
1 row created.
SQL> INSERT INTO SECRETARY VALUES(3,6,TO_DATE('17-MAR-2018','DD-MON-YYYY'),2);
1 row created.
SQL> INSERT INTO SECRETARY VALUES(4,8,TO_DATE('19-MAY-1996','DD-MON-YYYY'),1);
1 row created.
SQL> INSERT INTO SECRETARY VALUES(5,9,TO_DATE('7-FEB-1995','DD-MON-YYYY'),3);
1 row created.
SQL> INSERT INTO SECRETARY VALUES(6,10,TO_DATE('20-JAN-2000','DD-MON-YYYY'),1);
1 row created.
SQL> INSERT INTO JOB VALUES(1,'Emergency Valve Change',TO_DATE('01-JAN-2020','DD-MON-YYYY'));
1 row created.
SQL> INSERT INTO JOB VALUES(2,'Routine Faucet Inspection',TO_DATE('12-DEC-2019','DD-MON-YYYY'));
1 row created.
SQL> INSERT INTO JOB VALUES(3,'Routine Crystalliser Start-up',TO_DATE('23-MAR-2019','DD-MON-YYYY'));
1 row created.
SQL> INSERT INTO JOB VALUES(4,'Emergency Reactor Shutdown',TO_DATE('12-FEB-2021','DD-MON-YYYY'));
1 row created.
SQL> INSERT INTO JOB VALUES(5,'Emergency NO2 Leak Repair',TO_DATE('14-MAY-2019','DD-MON-YYYY'));
1 row created.
SQL> INSERT INTO JOB VALUES(6,'Routine Drum Drain',TO_DATE('15-OCT-2020','DD-MON-YYYY'));
1 row created.
SQL> INSERT INTO EMERGENCY_JOBS VALUES(1,1,TO_DATE('2020-JAN-01 12:30','YYYY-MON-DD HH24:MI'));
1 row created.
SQL> INSERT INTO EMERGENCY_JOBS VALUES(2,4,TO_DATE('2019-FEB-12 17:30','YYYY-MON-DD HH24:MI'));
1 row created.
SQL> INSERT INTO EMERGENCY_JOBS VALUES(3,5,TO_DATE('2019-MAY-14 10:00','YYYY-MON-DD
HH24:MI'));
1 row created.
```

```
SQL> INSERT INTO ROUTINE_JOBS VALUES(1,2);
1 row created.
SQL> INSERT INTO ROUTINE_JOBS VALUES(2,3);
1 row created.
SQL> INSERT INTO ROUTINE_JOBS VALUES(3,6);
1 row created.
SQL> INSERT INTO SITE VALUES(1,'Maydown',1);
1 row created.
SQL> INSERT INTO SITE VALUES(2,'Strathclyde',2);
1 row created.
SQL> INSERT INTO SITE VALUES(3,'Limivady',3);
1 row created.
SQL> INSERT INTO SITE VALUES(4,'Ringaskiddy',4);
1 row created.
SQL> INSERT INTO UNIT VALUES('A',1,200,'Light Industry',1);
1 row created.
SQL> INSERT INTO UNIT VALUES('B',2,320,'Heavy Industry',1);
1 row created.
SQL> INSERT INTO UNIT VALUES('C',3,150,'Distribution Centre',4);
1 row created.
SQL> INSERT INTO UNIT VALUES('D',4,100,'Hazardous Storage',1);
1 row created.
SQL> INSERT INTO UNIT VALUES('AB',5,100,'Light Industry',2);
1 row created.
SQL> INSERT INTO UNIT VALUES('BBC',2,300,'Hazardous Storage',2);
1 row created.
SQL> INSERT INTO UNIT VALUES('DA',6,50,'Heavy Industry',3);
1 row created.
SQL> INSERT INTO UNIT VALUES('F',1,100,'Hazardous Storage',4);
1 row created.
SQL> INSERT INTO JOBUNIT VALUES(1,'A',6);
1 row created.
SQL> INSERT INTO JOBUNIT VALUES(2,'BBC',5);
1 row created.
SQL> INSERT INTO JOBUNIT VALUES(3,'C',4);
1 row created.
SQL> INSERT INTO JOBUNIT VALUES(4,'B',3);
1 row created.
SQL> INSERT INTO JOBUNIT VALUES(5,'D',1);
1 row created.
SQL> INSERT INTO JOBUNIT VALUES(6,'DA',2);
1 row created.
```

4. Querying the Database

What are the biggest and smallest floor area for each unit type?

```
SQL> SELECT UNIT_NUMBER, MAX(FLOOR_AREA), MIN(FLOOR_AREA)
FROM UNIT, SITE
WHERE UNIT.SITE_NUMBER = SITE.SITE_NUMBER
AND FLOOR_AREA > 50
GROUP BY UNIT_NUMBER;
```

UNIT_NUMBER MAX(FLOOR_AREA) MIN(FLOOR_AREA)
---------- ---------- ----------
1 200 100
2 320 300
4 100 100
5 100 100
3 150 150

Which units have a floor area larger than the average unit floor area?

```
SQL> SELECT UNIT_ID, UNIT_NUMBER, FLOOR_AREA, ZONE, SITE_NUMBER
FROM UNIT
WHERE FLOOR_AREA > (SELECT AVG(FLOOR_AREA)
FROM UNIT);
```

UNIT_ID UNIT_NUMBER FLOOR_AREA ZONE SITE_NUMBER
---------- ---------- ---------- ---------- ----------
A 1 200 Light Industry 1
B 2 320 Heavy Industry 1
BBC 2 300 Hazardous Storage 2

Which units have a floor area larger than the average unit floor area for that type of unit?

```
SQL> SELECT *
  FROM UNIT
  WHERE FLOOR_AREA > (SELECT AVG(FLOOR_AREA)
  FROM UNIT X
  WHERE UNIT.UNIT_NUMBER = X.UNIT_NUMBER);
```

UNIT_ID UNIT_NUMBER FLOOR_AREA ZONE SITE_NUMBER
---------- ---------- ---------- ---------- ----------
A 1 200 Light Industry 1
B 2 320 Heavy Industry 1

```
SQL> SELECT
sec.SECRETARY_NUMBER,
sec.EMPLOYEE_NUMBER AS SEC_EMP,
emp.NAME AS SEC_NAME,
sec.MANAGER_NUMBER,
man.EMPLOYEE_NUMBER AS MAN_EMP,
emp2.NAME AS MAN_NAME
FROM SECRETARY sec
INNER JOIN EMPLOYEE emp
ON emp.EMPLOYEE_NUMBER = sec.EMPLOYEE_NUMBER
INNER JOIN MANAGER man
ON man.EMPLOYEE_NUMBER = sec.MANAGER_NUMBER
INNER JOIN EMPLOYEE emp2
ON emp2.EMPLOYEE_NUMBER = man.EMPLOYEE_NUMBER;
```

SECRETARY_NUMBER SEC_EMP SEC_NAME MANAGER_NUMBER MAN_EMP MAN_NAME
---------- ---------- ---------- ---------- ---------- ----------
4 8 Jack Hackett 1 1 Charlie McKerracher
6 10 Wade Wilson 1 1 Charlie McKerracher
2 4 Ian McElhinney 3 3 Colm Meaney
5 9 Hugh Jackman 3 3 Colm Meaney

```
SQL> SELECT
men.Mentor_ID,
```

```
man.Employee_Number AS MENTOR_EMP,
emp.Name,
men.Mentee_ID,
man2.Employee_Number AS MENTEE_EMP,
emp2.Name
FROM MENTOR men
LEFT JOIN MANAGER man
ON men.MENTOR_ID = man.MANAGER_NUMBER
LEFT JOIN MANAGER man2
ON men.MENTEE_ID = man2.MANAGER_NUMBER
LEFT JOIN EMPLOYEE emp
ON man. EMPLOYEE_NUMBER = emp. EMPLOYEE_NUMBER
LEFT JOIN EMPLOYEE emp2
ON man2. EMPLOYEE_NUMBER = emp2. EMPLOYEE_NUMBER;
```

MENTOR_ID	MENTOR_EMP NAME	MENTEE_ID	MENTEE_EMP NAME
2	3 Colm Meaney	1	1 Charlie McKerracher
4	7 Ted Crilly	3	5 Tommy Tiernan

5. Critique of your database

My database is strong as it adheres strongly to third normal form, with each table containing attributes that are dependent only on the primary key, with the addition of foreign keys in some of the tables (such as manager_number in the site table) which represent the relationships between tables. There are also strong integrity constraints which prevent null values being entered in key columns, as well as enforcing the relationship between tables, meaning for instance that a secretary cannot be added to the secretary table with an employee_number that is not already present in the employee table. This can also be a weakness though, as it requires that any new data has to be added to the tables in the correct order, or it will be rejected by the database. Adhering to 3NF also required extra tables to be created, in order to ensure that there were no transitive dependencies present in the database, however this comes at the cost of making the overall database much more complex. In this instance though the database may have been better kept in 2NF as this makes it much more intuitive for a human user, albeit at the cost of increased difficulty in machine processing.

When creating the table, I began by creating a table for each entity, including all of the relevant attributes for each one. I then added foreign keys to tables that contained non-obligatory 1:1 relationships or obligatory 1:N relationships, such as inserting Site_Number into the Unit table. In order to accurately represent the M:N relationships present in the EER Diagram (such as JOBUNIT) I added a linker table containing both Job_Number and Unit_ID, as well as a Ju_Number as a primary key. Similarly to represent the Mentors relationship between different managers I created a Mentor table containing a Mentor_ID and a Mentee_ID, with Mentee_ID as the primary key since a mentee is only mentored by one mentor but a single mentor might have several mentees. The functional dependencies are shown below:

SITE: Site_Number -> Name, Manager_ID
UNIT: Unit_ID -> Unit_Number, Floor_Area, Zone, Site_Number
EMPLOYEE: Employee_Number -> Name, Job_Title, Home_Address, Telephone_Number
MANAGER: Manager_Number -> Employee_Number
MENTOR: Mentee_ID -> Mentor_ID
SECRETARY: Secretary_Number -> Employee_Number, Start_Date, Manager_Number
JOB: Job_Number -> Description, Logdate
JOBUNIT: Ju_Number -> Unit_ID, Job_Number
EMERGENCY_JOBS: E_Number -> Job_Number, E_Date_and_Time)
ROUTINE_JOBS: R_Number -> Job_Number)