1 Logical Design

1.1 Subtype Representation

1.1.1 Staff Tables

$$\texttt{STAFF(\underline{Esnum},Ephone,StartingDate)};\ \texttt{TEACHER(\underline{TeachID},Qualification,Grade)};\ \texttt{TECHNICIAN(\underline{TechNo})}$$

$$\therefore \quad \left.\begin{array}{r}\text{Staff (Parent)}\\ \text{Teacher (Child)}\\ \text{Technician (Child)}\end{array}\right\} \begin{array}{l}\texttt{TEACHER(\underline{TeachID}, Ename, Ephone, StartingDate, Qualification, Grade)}\\ \texttt{TECHNICIAN(\underline{TechNo}, Ename, Ephone, StartingDate)}\end{array}$$

1.1.2 Module Tables

$$\texttt{MODULE(\underline{ModCode},Title,Credits)};\ \texttt{INPERSON(\underline{PID},Lectures)};\ \texttt{ONLINE(\underline{OID},ContentHours)}$$

$$\therefore \quad \left.\begin{array}{r}\text{Module (Parent)}\\ \text{InPerson (Child)}\\ \text{OnLine (Child)}\end{array}\right\} \begin{array}{l}\texttt{INPERSON(\underline{PID}, Title, Credits, Lectures)}\\ \texttt{ONLINE(\underline{OID}, Title, Credits, ContentHours)}\end{array}$$

1.2 Relationship Representation

1. **PresentedBy** (Module, Teacher): all attributes to existing table
2. **References** (InPerson, OnLine): new table for relationship
3. **TaughtIn** (InPerson, Room): identifier of ‘one’ table to ‘many’ table
4. **Supervises** (Technician, Technician): new table for relationship

1.3 Final Tables

- TEACHER(TeachID, Ename, Ephone, StartingDate, Qualification, Grade)
- TECHNICIAN(TechNo, Ename, Ephone, StartingDate)
- INPERSON(PID, Title, Credits, Lectures, TeachID[1], RoomNumber[3])
- ONLINE(OID, Title, Credits, ContentHours, TeachID[1])
- ROOM(RoomNumber, Seats, DataProjector)
- MODULE_REF(OID, PID)[2]
- SUPERVISION(TechNo, TechNo)[4]

2 SQL: Table Construction

```
CREATE TABLE teacher (
         tr_id           CHAR(3) CONSTRAINT pk_trid PRIMARY KEY,
         tr_name         VARCHAR(20),
         tr_phone        CHAR(11) CONSTRAINT tr_phone_valid CHECK (tr_phone NOT LIKE '%[^0-9]%'),
         tr_start_date   DATE,
         qualification   VARCHAR(20),
         grade           VARCHAR(10)
);

CREATE TABLE technician (
         tn_id           CHAR(3) CONSTRAINT pk_tnid PRIMARY KEY,
         tn_name         VARCHAR(20),
         tn_phone        CHAR(11) CONSTRAINT tn_phone_valid CHECK (tn_phone NOT LIKE '%[^0-9]%'),
         tn_start_date   DATE
);

CREATE TABLE in_person (
         p_id            CHAR(5) CONSTRAINT pk_pid PRIMARY KEY,
         p_title         VARCHAR(20),
         p_credits       NUMBER(3),
         nlectures       VARCHAR(2),
         tr_id           CHAR(3) CONSTRAINT p_tr_id_ref REFERENCES teacher(tr_id),
         room_no         CHAR(5) CONSTRAINT room_no_ref REFERENCES room(room_no)
);

CREATE TABLE on_line (
        o_id            CHAR(5) CONSTRAINT pk_oid PRIMARY KEY,
        o_title         VARCHAR(20),
        o_credits       NUMBER(3),
        ncontent_hours  VARCHAR(3),
        tr_id           CHAR(3) CONSTRAINT o_tr_id_ref REFERENCES teacher(tr_id)
);

CREATE TABLE room (
        room_no         CHAR(5) CONSTRAINT pk_room PRIMARY KEY,
        seats           NUMBER(3),
        data_projector  CHAR(1) CONSTRAINT proj_valid CHECK (data_projector IN('Y', 'N'))
);

CREATE TABLE module_ref (
        o_id            CHAR(5) CONSTRAINT o_id_ref REFERENCES on_line(o_id),
        p_id            CHAR(5) CONSTRAINT p_id_ref REFERENCES in_person(p_id)
);

CREATE TABLE supervision (
         tn_id_sd        CHAR(3) CONSTRAINT tn_id_sd_ref REFERENCES technician(tn_id),
         tn_id_sr        CHAR(3) CONSTRAINT tn_id_sr_ref REFERENCES technician(tn_id)
);
```

`Table created.` ([7,1]) for 7 tables and one output per table.

3 SQL: Table Population

```
INSERT INTO teacher VALUES(’T01’, ’Federer, Roger’, ’01416382226’, ’23-MAY-1984’, ’PhD Vimtutor’, ’Professor’);
INSERT INTO teacher VALUES(’T02’, ’Nadal, Rafael’, ’01416385638’, ’12-FEB-1987’, ’PhD GNU Troff’, ’Lecturer’);
INSERT INTO teacher VALUES(’T03’, ’Djokovic, Novak’, ’01416383947’, ’09-MAR-1989’, ’PhD Lindypress’, ’Lecturer’);
INSERT INTO teacher VALUES(’T04’, ’Sawyer, Peyton’, ’01416380195’, ’21-JUL-1996’, ’PhD Lucas Scott’, ’Reader’);
INSERT INTO teacher VALUES(’T05’, ’Davis, Brooke’, ’01416385720’, ’27-APR-1992’, ’PhD A6 allroads’, ’Reader’);
INSERT INTO teacher VALUES(’T06’, ’Collins, Lily’, ’01416389437’, ’14-DEC-1998’, ’PhD Vanlife’, ’Professor’);
INSERT INTO teacher VALUES(’T07’, ’Grow, Custom’, ’01416383021’, ’01-APR-1982’, ’PhD Mutt’, ’Lecturer’);
INSERT INTO teacher VALUES(’T08’, ’Henderson, Graham’, ’01416380763’, ’23-AUG-1991’, ’PhD neoMutt’, ’Lecturer’);
INSERT INTO teacher VALUES(’T09’, ’Munro, Hugh’, ’01416385950’, ’21-JUN-1999’, ’PhD Beinn Chabhair’, ’Lecturer’);
INSERT INTO teacher VALUES(’T10’, ’Castillo, Martin’, ’01416389002’, ’19-FEB-1993’, ’PhD Bowls’, ’Professor’);

INSERT INTO technician VALUES(’M01’, ’Crockett, James’, ’01416380192’, ’11-DEC-1999’);
INSERT INTO technician VALUES(’M02’, ’Tubbs, Ricardo’, ’01416388704’, ’03-SEP-2002’);
INSERT INTO technician VALUES(’M03’, ’Moolenaar, Bram’, ’01416386696’, ’08-AUG-1987’);
INSERT INTO technician VALUES(’M04’, ’Sullivan, Victor’, ’01416386966’, ’12-SEP-1963’);
INSERT INTO technician VALUES(’M05’, ’Drake, Nathan’, ’01416388675’, ’14-MAY-1981’);
INSERT INTO technician VALUES(’M06’, ’Smith, Luke’, ’01416388156’, ’28-JUN-1801’);
INSERT INTO technician VALUES(’M07’, ’Diesel, Vim’, ’01416386017’, ’08-NOV-1984’);
INSERT INTO technician VALUES(’M08’, ’All, David’, ’01416389119’, ’12-DEC-1969’);
INSERT INTO technician VALUES(’M09’, ’Road, Davey’, ’01416388609’, ’14-MAR-1982’);
INSERT INTO technician VALUES(’M10’, ’Wealth, Money’, ’01416389010’, ’28-JUL-1991’);

INSERT INTO in_person VALUES(’CS626’, ’Advanced Vimtutor’, 120, 12, ’T06’, ’M105’);
INSERT INTO in_person VALUES(’CS628’, ’Advanced Crampon’, 160, 20, ’T01’, ’M103’);
INSERT INTO in_person VALUES(’CS665’, ’Advanced CustomGrow’, 120, 12, ’T03’, ’M102’);
INSERT INTO in_person VALUES(’CS696’, ’Advanced Vanlife’, 140, 16, ’T02’, ’M108’);
INSERT INTO in_person VALUES(’CS698’, ’Advanced Model M’, 140, 14, ’T05’, ’M108’);
INSERT INTO in_person VALUES(’CS705’, ’Advanced Clachaig’, 120, 12, ’T04’, ’M108’);

INSERT INTO on_line VALUES(’CS428’, ’Advanced Chess’, 140, 36, ’T08’);
INSERT INTO on_line VALUES(’CS435’, ’Advanced Topspin’, 140, 36, ’T10’);
INSERT INTO on_line VALUES(’CS454’, ’Advanced TCR A1’, 140, 36, ’T07’);
INSERT INTO on_line VALUES(’CS458’, ’Advanced allroad’, 140, 36, ’T09’);

INSERT INTO room VALUES(’M101’, 115, ’N’);
INSERT INTO room VALUES(’M102’, 205, ’Y’);
INSERT INTO room VALUES(’M103’, 180, ’Y’);
INSERT INTO room VALUES(’M104’, 165, ’N’);
INSERT INTO room VALUES(’M105’, 190, ’Y’);
INSERT INTO room VALUES(’M106’, 210, ’Y’);
INSERT INTO room VALUES(’M107’, 95, ’N’);
INSERT INTO room VALUES(’M108’, 230, ’Y’);

INSERT INTO module_ref VALUES(’CS435’, ’CS626’);
INSERT INTO module_ref VALUES(’CS454’, ’CS628’);
INSERT INTO module_ref VALUES(’CS458’, ’CS665’);
INSERT INTO module_ref VALUES(’CS435’, ’CS696’);
INSERT INTO module_ref VALUES(’CS454’, ’CS698’);
INSERT INTO module_ref VALUES(’CS435’, ’CS705’);
INSERT INTO module_ref VALUES(’CS454’, ’CS705’);

INSERT INTO supervision VALUES(’M03’, ’M01’);
INSERT INTO supervision VALUES(’M04’, ’M01’);
INSERT INTO supervision VALUES(’M05’, ’M01’);
INSERT INTO supervision VALUES(’M06’, ’M01’);
INSERT INTO supervision VALUES(’M07’, ’M02’);
INSERT INTO supervision VALUES(’M08’, ’M02’);
INSERT INTO supervision VALUES(’M09’, ’M02’);
INSERT INTO supervision VALUES(’M10’, ’M02’);
```

`1 row created.` ([7,N]) for 7 tables and N outputs per table relative to N inputs per table.

4 SQL: Queries

(a) Find the maximum number of lectures and minimum number of lectures relative to classes thaught in each used room. Therefore, grouped by room number. In this case, only M108 is used more than once and the number of lectures for the classes which use the room vary, so there is a valid maximum and minimum.

```
COLUMN room_no HEADING ’Room Number’ FORMAT A15
COLUMN MAX(nlectures) HEADING ’Max N Lectures’ FORMAT A20
COLUMN MIN(nlectures) HEADING ’Min N Lectures’ FORMAT A20

SELECT room.room_no, MAX(nlectures), MIN(nlectures)
        FROM in_person, room
        WHERE in_person.room_no = room.room_no
        GROUP BY room.room_no;

Room Number     Max N Lectures       Min N Lectures
--------------- -------------------- --------------------
M102            12                   12
M105            12                   12
M103            20                   20
M108            16                   12
```

(b) Find the classes which are taught in the room with the maximum number of of seats. This is executed by finding the max number in the seats column of the room table and the correlating room number, and displaying classes taught in this room and the corresponding teachers.

```
COLUMN p_id HEADING ’Class ID’ FORMAT A10
COLUMN p_title HEADING ’Class Name’ FORMAT A20
COLUMN nlectures HEADING ’N Lectures’ FORMAT A15
COLUMN tr_name HEADING ’Taught By’ FORMAT A15

SELECT p_id, p_title, nlectures, tr_name
        FROM in_person, teacher, room
        WHERE in_person.tr_id = teacher.tr_id
        AND room.room_no = in_person.room_no
        AND room.seats = (
                SELECT MAX(seats)
                FROM room
        );

Class ID   Class Name           N Lectures      Taught By
---------- -------------------- --------------- ---------------
CS696      Advanced Vanlife     16              Nadal, Rafael
CS698      Advanced Model M     14              Davis, Brooke
CS705      Advanced Clachaig    12              Sawyer, Peyton
```

(c) For the most occuring used room, find the classes with a number of credits which are above the average for the classes taught in that room. This may not be considered particularly ‘relevant’ or ‘useful’ however, it’s certainly something I would be contemplating back in a first year JA325 finance lecture.

```
COLUMN p_id HEADING ’Class ID’ FORMAT A10
COLUMN p_title HEADING ’Class Title’ FORMAT A20
COLUMN p_credits HEADING ’Credits’
COLUMN room_no HEADING ’Room Number’ FORMAT A15
```

```
SELECT p_id, p_title, p_credits, room_no
         FROM in_person
         WHERE p_credits > (
                 SELECT AVG(p_credits)
                         FROM in_person ip
                         WHERE in_person.room_no = ip.room_no
         );

Class ID    Class Title              Credits  Room Number
----------  --------------------  ----------  ---------------
CS696       Advanced Vanlife             140  M108
CS698       Advanced Model M             140  M108
```

(d)

```
COLUMN o_title HEADING 'Online Class (Refd)' FORMAT A23
COLUMN ncontent_hours HEADING 'Online Content Hrs' FORMAT A15
COLUMN p_title HEADING 'In-Person Class (Refr)' FORMAT A23

SELECT o_title, ncontent_hours, p_title
        FROM in_person, on_line, module_ref
        WHERE in_person.p_id = module_ref.p_id
        AND module_ref.o_id = on_line.o_id;

Online Class (Refd)      Online Content   In-Person Class (Refr)
-----------------------  ---------------  -----------------------
Advanced Topspin         36               Advanced Vimtutor
Advanced TCR A1          36               Advanced Crampon
Advanced allroad         36               Advanced CustomGrow
Advanced Topspin         36               Advanced Vanlife
Advanced TCR A1          36               Advanced Model M
Advanced Topspin         36               Advanced Clachaig
Advanced TCR A1          36               Advanced Clachaig
```

(e)

```
COLUMN supd.tn_name FORMAT A20
COLUMN supr.tn_name FORMAT A20

SELECT supd.tn_name "Supervised Name", supr.tn_name "Supervisor Name"
         FROM technician supr, technician supd, supervision
         WHERE supd.tn_id = supervision.tn_id_sd
         AND supr.tn_id = supervision.tn_id_sr;

Supervised Name       Supervisor Name
--------------------  --------------------
Moolenaar, Bram       Crockett, James
Sullivan, Victor      Crockett, James
Drake, Nathan         Crockett, James
Smith, Luke           Crockett, James
Diesel, Vim           Tubbs, Ricardo
All, David            Tubbs, Ricardo
Road, Davey           Tubbs, Ricardo
Wealth, Money         Tubbs, Ricardo
```

5 Design & Implementation Critique

This database structure effectively segregates relevant data into approporiate tables which are references and joined appropriately in the given scenarios. Separating the teacher and technician tables gathered approporiate data allowing dependency upon the relevant type of staff ID, not just ggregate staff ID. That is, functional dependencies as follow: `TeachID` → `Ename, Ephone, StartingDate, Qualification, Grade`; `TechNo` → `Ename, Ephone, StartingDate`, with each repeating attribute type respective to their unique staff type. This made the creation of the Supervision table and query (e) more accessible. Likewise, the same case exists with the module scenario. Where the separation into InPerson and OnLine, with dependencies: `PID` → `Title, Credits, Lectures`; `OID` → `Title, Credits, ContentHours`, also made the creation of the ModuleRef table and query (d) more accessible. Once, again column names are alike however as seen, claim different more relevant titles in practice. Of course, this leaves the remaining dependency: `RoomNumber` → `Seats, DataProjector`.

On the other hand, the final two queries (d) and (e) are partially fulfilling of the required criteria. That is, non-referenced online modules do not appear in the results of query (d) and non-supervising technicians do not appear in the same column as those who supervise in the results of query (e). Therefore, a penalty is recognised here. On a slightly less important note, an imporvement could have been made to both the teacher and technician tables where the Ename fields are populated with both the individual's forename preceeded by their surname; comma-separated. In line with normal form convention, forename and surname could have been allocated their own fields however, it may have been irrelevant to do so as in this case forenames always appear uniquely with the appropriate according surnames.

Critique Word Count: 284