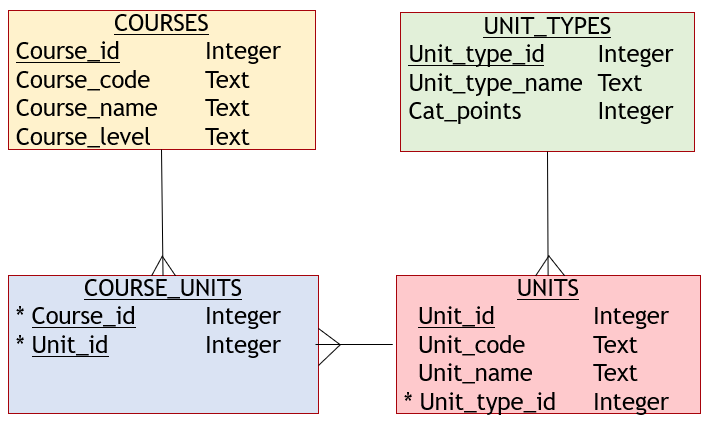
**POPULATING A DATABASE – PRACTICE**

Below is our design for the course unit database which we will now implement:



1. Creating the tables

Make sure FOREIGN KEY constraints are enabled in SQLite by entering the following command:

PRAGMA foreign\_keys = ON;

The tables have to be created in the right order. Firstly the tables that don’t reference other tables i.e. don’t have any foreign keys – Courses and Unit\_types. Then you can create the Units table next and finally the Course\_units table.

CREATE TABLE unit\_types

(unit\_type\_id INTEGER PRIMARY KEY AUTOINCREMENT,

unit\_type\_name TEXT UNIQUE NOT NULL,

cat\_points INTEGER

);

CREATE TABLE units

(unit\_id INTEGER PRIMARY KEY AUTOINCREMENT,

unit\_code TEXT UNIQUE NOT NULL,

unit\_name TEXT NOT NULL,

unit\_type\_id INTEGER NOT NULL,

CONSTRAINT unit\_type\_fk FOREIGN KEY (unit\_type\_id)

REFERENCES unit\_types(unit\_type\_id)

);

CREATE TABLE courses

(course\_id INTEGER PRIMARY KEY AUTOINCREMENT,

course\_code TEXT UNIQUE NOT NULL,

course\_name TEXT NOT NULL,

course\_level TEXT NOT NULL,

CONSTRAINT course\_level\_check CHECK (course\_level IN ('MSc','BSc','HND'))

);

CREATE TABLE course\_units

(course\_id INTEGER,

unit\_id INTEGER,

CONSTRAINT course\_units\_pk PRIMARY KEY (course\_id, unit\_id),

CONSTRAINT course\_id\_fk FOREIGN KEY (course\_id)

REFERENCES courses(course\_id),

CONSTRAINT unit\_id\_fk FOREIGN KEY (unit\_id)

REFERENCES units(unit\_id)

);

1. Altering the design

After coming up with the design, we have found that we have missed out an attribute from the COURSES table to store the length of the course.

Use the following command to ALTER the table to add this column:

ALTER TABLE courses

ADD COLUMN course\_length INTEGER;

We then discover that there should have been a CHECK constraint on the course\_length column to only allow values between 1 and 4 but as the SQLite ALTER TABLE command does not provide a way to add a constraint, we will have to drop the table and recreate it correctly as follows:

DROP TABLE course\_units; -- This is required as we cant drop the course table until all

-- tables that reference it are dropped.

DROP TABLE courses;

CREATE TABLE courses

(course\_id INTEGER PRIMARY KEY AUTOINCREMENT,

course\_code TEXT UNIQUE NOT NULL,

course\_name TEXT NOT NULL,

course\_level TEXT NOT NULL,

course\_length INTEGER NOT NULL,

CONSTRAINT course\_level\_check CHECK (course\_level IN ('MSc','BSc','HND')),

CONSTRAINT course\_length\_check CHECK (course\_length between 1 and 4)

);

CREATE TABLE course\_units

(course\_id INTEGER,

unit\_id INTEGER,

CONSTRAINT course\_units\_pk PRIMARY KEY (course\_id, unit\_id),

CONSTRAINT course\_id\_fk FOREIGN KEY (course\_id)

REFERENCES courses(course\_id),

CONSTRAINT unit\_id\_fk FOREIGN KEY (unit\_id)

REFERENCES units(unit\_id)

);

1. Inserting data

Copy and paste the following commands to populate the database:

INSERT INTO unit\_types (unit\_type\_name, cat\_points)

VALUES ('Basic',10);

INSERT INTO unit\_types (unit\_type\_name, cat\_points)

VALUES ('Intermediate',20);

INSERT INTO unit\_types (unit\_type\_name, cat\_points)

VALUES ('Advanced',30);

INSERT INTO courses (course\_code, course\_name, course\_level, course\_length)

VALUES ('B74','Comp Science','BSc', 3);

INSERT INTO courses (course\_code, course\_name, course\_level, course\_length)

VALUES ('B94','Comp Applications','MSc', 2);

INSERT INTO units (unit\_code, unit\_name, unit\_type\_id)

VALUES ('B741','Program 1',1);

INSERT INTO units (unit\_code, unit\_name, unit\_type\_id)

VALUES ('B742','Hardware 1',1);

INSERT INTO units (unit\_code, unit\_name, unit\_type\_id)

VALUES ('B743','Data Processing 1',1);

INSERT INTO units (unit\_code, unit\_name, unit\_type\_id)

VALUES ('B744','Program 2',2);

INSERT INTO units (unit\_code, unit\_name, unit\_type\_id)

VALUES ('B745','Hardware 2',2);

INSERT INTO units (unit\_code, unit\_name, unit\_type\_id)

VALUES ('B951','Information',3);

INSERT INTO units (unit\_code, unit\_name, unit\_type\_id)

VALUES ('B952','Microprocessors',3);

INSERT INTO course\_units (course\_id, unit\_id)

VALUES (1,1); -- 'B74','B741'

INSERT INTO course\_units (course\_id, unit\_id)

VALUES (1,2); -- 'B74','B742'

INSERT INTO course\_units (course\_id, unit\_id)

VALUES (1,3); -- 'B74','B743'

INSERT INTO course\_units (course\_id, unit\_id)

VALUES (1,4); -- 'B74','B744'

INSERT INTO course\_units (course\_id, unit\_id)

VALUES (1,5); -- 'B74','B745'

INSERT INTO course\_units (course\_id, unit\_id)

VALUES (2,6); -- 'B94','B951'

INSERT INTO course\_units (course\_id, unit\_id)

VALUES (2,7); -- 'B94','B952'

INSERT INTO course\_units (course\_id, unit\_id)

VALUES (2,1); -- 'B94','B741'

Select from each of the four tables to check that the data has inserted correctly. Notice we did not insert data into the primary key columns but these were automatically populated by a SQLite sequence created by the AUTOINCREMENT clause on the CREATE TABLE commands.

1. Updating and deleting data

Update the Unit table and change the name of unit B741 to ‘Programming 1’

UPDATE units

SET unit\_name = 'Programming 1'

WHERE unit\_code = 'B741';

Update the Unit table and change the name of unit B744 to ‘Programming 2’

UPDATE units

SET unit\_name = 'Programming 2'

WHERE unit\_code = 'B744';

Update the Course table and do a global replace of ‘Comp’ with ‘Computer’ in the course name column.

UPDATE courses

SET course\_name = 'Computer '||SUBSTR(course\_name,6,LENGTH(course\_name)-5)

WHERE course\_name like 'Comp%';

The ‘Hardware 2’ unit is no longer being run so remove it from all courses that teach it and then remove the unit itself. Assume you don’t know the unit id or unit code.

DELETE FROM course\_units

WHERE unit\_id = (SELECT unit\_id

FROM units

WHERE unit\_name = 'Hardware 2');

DELETE FROM units

WHERE unit\_name = 'Hardware 2';

1. Testing the constraints
2. NOT NULL constraint

INSERT INTO courses (course\_code, course\_level)

VALUES ('B84','BSc');

Should generate a NOT NULL constraint failed: course.course\_name error as a NULL value is not allowed in the course\_name column.

1. UNIQUE constraint

INSERT INTO units (unit\_id, unit\_code,unit\_name, unit\_type\_id)

VALUES (7, 'B951', 'Data Analysis', 2);

Should generate a UNIQUE constraint failed: units.unit\_id error as a unit id of 7 already exists and it’s a primary key.

1. FOREIGN KEY constraint

INSERT INTO course\_units

VALUES (3,1);

Should generate a FOREIGN KEY constraint failed error as the course\_id 3 doesn’t exist in the Courses table.

DELETE FROM units

WHERE unit\_code = 'B741';

Should generate a FOREIGN KEY constraint failed error as the unit\_id of 1 which relates to unit\_code B741 exists in the Course\_units table.

1. CHECK constraint

UPDATE courses

SET course\_length = 5

WHERE course\_code = 'B74';

Should generate a CHECK constraint failed: course\_length\_check as the course\_length must be between 1 and 4.