

No repeating groups in the table.

UNF

UNITS(unit\_number, unit\_name, unit\_description\_unit\_value)

1NF

* Identify the composite keys (a unique id). Hence, candidate keys are:
* Unit\_number
* Unit\_name (can have the same name)
* Unit\_description (can have the same description)

UNITS(unit\_number, unit\_name, unit\_description\_unit\_value)

**Partial dependency (you look at the candidate keys for partial dependences)**

To get from 1NF to 2NF, you need to find partial dependencies (one of the attribute in the composite key determines a non-key attribute).

* None

2NF

UNIT(unit\_number, unit\_name, unit\_description\_unit\_value)

**Transitive Dependency**  
**non-key is everything except in candidate keys**

To go from 2NF -> 3NF we use transitive dependencies (non-key attribute depends on another non-key attribute, which in turn depends on the primary key).

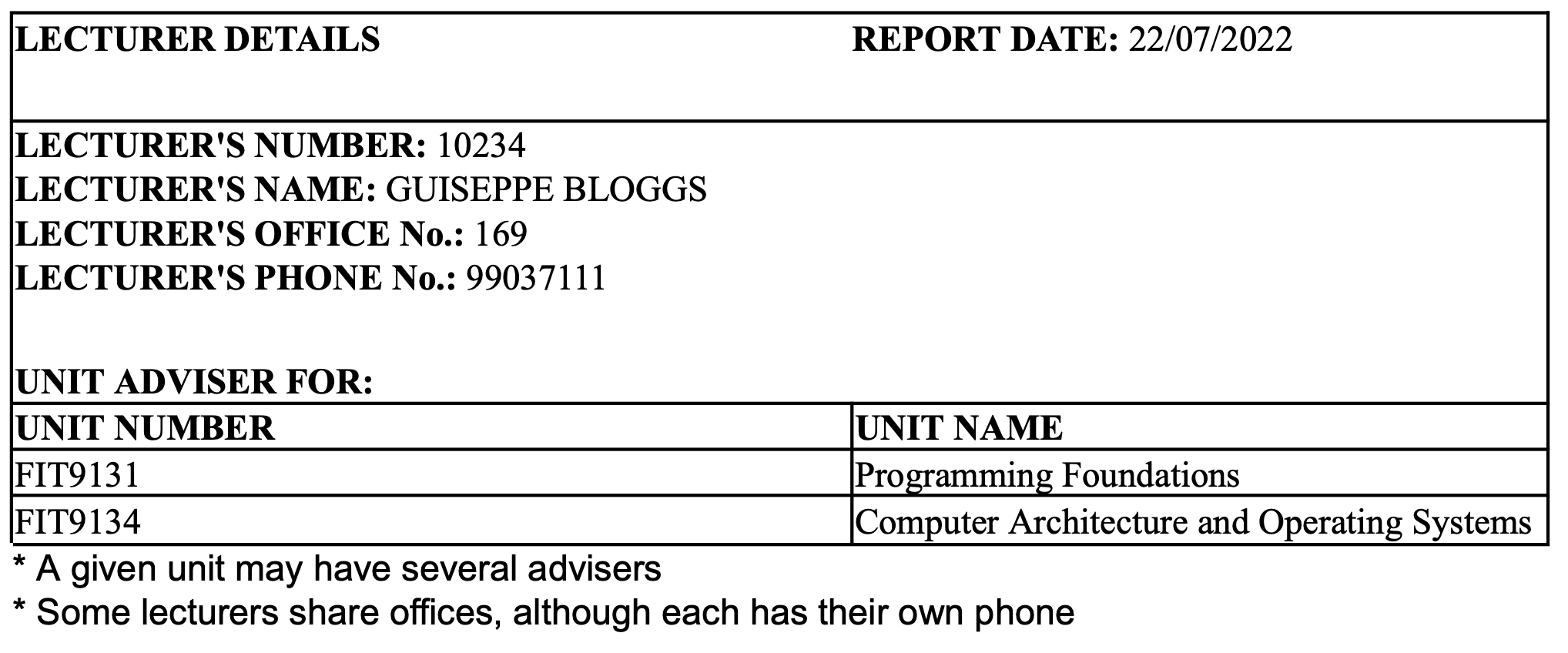
Transitive Dependency

* None

UNIT(unit\_number, unit\_name, unit\_description\_unit\_value)

Full dependencies

Unit\_number -> unit\_name, unit\_description, unit\_value



Repeating group is unit\_number and unit\_name.

unit\_name can repeat, so it is not a composite key.

UNF

LECTURER\_DETAILS(lecture\_number, **lecture\_frame, lectire\_lname**, lecture\_office\_no, lecture\_phone\_no, (unit\_number, unit\_name))

1NF

LECTURER\_DETAILS(lecture\_number, **lecture\_frame, lectire\_lname**, lecture\_office\_no, lecture\_phone\_no, (unit\_number, unit\_name))

* (1) Remove repeating group
  + LECTURER\_DETAILS(lecture\_number, **lecture\_frame, lectire\_lname**, lecture\_office\_no, lecture\_phone\_no, (unit\_number, unit\_name)) ->
    - LECTURER\_DETAILS(lecture\_number, **lecture\_frame, lectire\_lname**, lecture\_office\_no, lecture\_phone\_no)
    - UNIT(unit\_number, unit\_name)
  + Candidate keys
    - Lecture\_no
    - Lecture\_phone
    - ~~Lecture\_office~~
    - Lecture\_no, unit\_number
  + Integrate candidate key into repeating group relation (we choose lecture\_no as PK)
    - LECTURER\_DETAILS(lecture\_number, **lecture\_frame, lectire\_lname**, lecture\_office\_no, lecture\_phone\_no)
    - LECTURER\_UNIT(lecture\_number, unit\_number, unit\_name)
  + Note: PK is not unique, because unit can have multiple lecturers. Hence, unit\_no is not enough to uniquely identify it. We also want lecture\_no as PK as well. Thus, we take lecturer\_name as a PK to make it composite and uniquely identifiable.

Partial dependencies **occurs when one primary key determines some other attribute/attributes**:

~~lecture\_number - > lecture\_name~~

Unit\_number -> unit\_name

2NF

LECTURER(lecture\_number, lecture\_frame, lectire\_lname, lecture\_office\_no, lecture\_phone\_no)

LECTURER\_UNIT(lecture\_number, unit\_number)

UNIT(unit\_number, unit\_name)

Transitive dependencies **occurs when some non-key attribute determines some other attribute**:

None

3NF

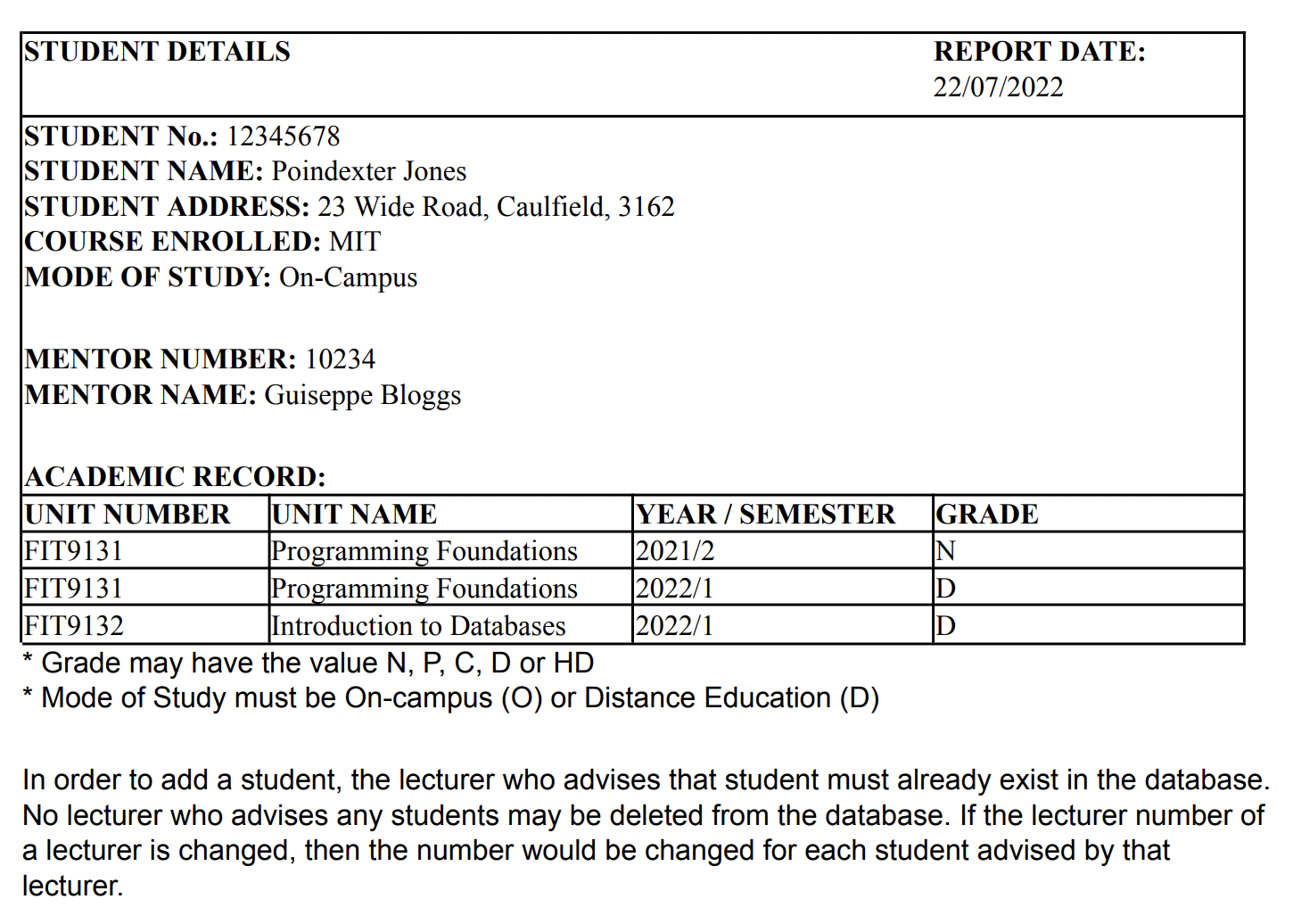
LECTURER(lecture\_number, lecture\_frame, lectire\_lname, lecture\_office\_no, lecture\_phone\_no)

LECTURER\_UNIT(lecture\_number, unit\_number)

UNIT(unit\_number, unit\_name)

Full dependencies **occurs when the full key is required (all columns of the key) to determine another attribute**:

* lecture\_number -> lecture\_fname, lecture\_lname, lecture\_office\_no, lecture\_phone
* unit\_number -> unit\_name



Repeating group is academic record

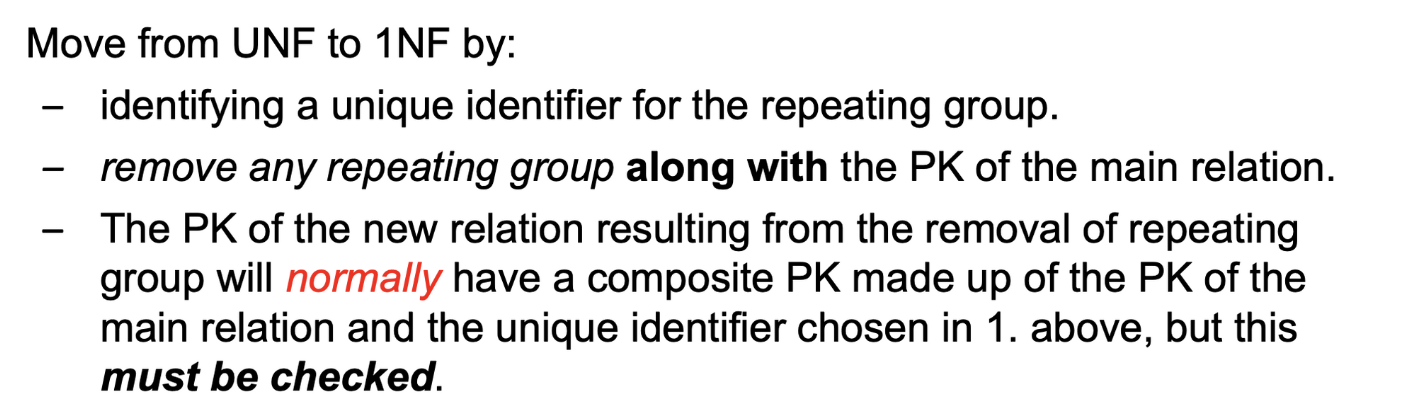
STUDENT\_DETAILS(student\_no, student\_name, student\_address, course\_enrolled, more\_of\_study, mentor\_number, mentor\_name, unit\_number, unit\_name, year, grade)

**Decompose student\_name into student\_fname and student\_lname**

STUDENT\_DETAILS(student\_no, student\_fname, student\_lname, student\_address, course\_enrolled, more\_of\_study, mentor\_number, mentor\_name, unit\_number, unit\_name, year, grade)

UNF

STUDENT\_DETAILS(student\_no, student\_fname, student\_lname, student\_address, course\_enrolled, more\_of\_study, mentor\_number, mentor\_name, (unit\_number, unit\_name, year, grade))



UNF -> 1NF:

STUDENT\_DETAILS(student\_no, student\_fname, student\_lname, student\_address, course\_enrolled, more\_of\_study, mentor\_number, mentor\_name)

ACADEMIC\_RECORD(student\_no, unit\_number, unit\_year, unit\_sem, unit\_name, grade)

Don’t combine candidate key from STUDENT with repeating groups.

Candidate keys:

student\_no

Student\_no,unit\_no, unit\_year, unit\_sem

**Partial dependency (you look at the candidate keys for partial dependences)**

To get from 1NF to 2NF, you need to find partial dependencies (one of the attribute in the composite key determines a non-key attribute).

* Unit\_number -> unit\_name

2NF

STUDENT\_DETAILS(student\_no, student\_fname, student\_lname, student\_address, course\_enrolled, more\_of\_study, mentor\_number, mentor\_name)

ACADEMIC\_RECORD(student\_no, unit\_number, unit\_year, unit\_sem, unit\_name, grade)

UNIT(unit\_no, unit\_name)

**Transitive Dependency**

To go from 2NF -> 3NF we use transitive dependencies (non-key attribute depends on another non-key attribute, which in turn depends on the primary key).

Transitive Dependency

* lecture\_no -> lecture\_fname, lecture\_lname

3NF

STUDENT\_DETAILS(student\_no, student\_fname, student\_lname, student\_address, course\_enrolled, more\_of\_study, mentor\_number, mentor\_name)

ACADEMIC\_RECORD(student\_no, unit\_number, unit\_year, unit\_sem, unit\_name, grade)

UNIT(unit\_no, unit\_name)

LECTURER(lecture\_no, lecture\_fname, lecture\_lname)

**Full dependencies**

student\_no -> student\_fname, student\_lname, student\_address, course\_enrolled, more\_of\_study, mentor\_number, mentor\_name)

student\_no, unit\_number -> unit\_year, unit\_sem, unit\_name, grade)

unit\_no, -> unit\_name

lecture\_no -> lecture\_fname, lecture\_lname

**Relations**

1. STUDENT(student no, student fame, student Iname, student address, student course, student mode, lecture\_no)

2. ACADEMIC\_RECORD(student no, unit no, unit year, unit sem, unit grade)

3. UNIT (unit no, unit name)

4. LECTURER(lecture no, lecture fname, lecture \_ Iname)

5. LECTURER(lecture no, lecture \_ fname, lecture\_ Iname, lecture\_office\_no, lecture phone)

6. LECTURER\_UNIT(lecture no, unit no)

7. UNIT (unit no ‚unit name)

8. UNIT(unit no, unit name, unit description, unit value)

**Final Relations:**

3,7&8: UNIT(unit no, unit name, unit [description, unit\_value)

4&5 : LECTURER(lecture no, lecture\_ fame, lecture Iname, lecture\_ office \_no, lecture phone)

LECTURER UNIT(lecture no, unit no)

STUDENT(student no, student fame, student Iname, student address, student\_ course, student mode, lecture no)

ACADEMIC\_RECORD(student\_no, unit\_no, unit\_year, unit\_sem, unit\_grade)

Attribute synthesis

collect all similar relations

1. UNIT(unit\_number, unit\_name, unit\_description\_unit\_value)
2. LECTURER(lecture\_number, lecture\_frame, lectire\_lname, lecture\_office\_no, lecture\_phone\_no)
3. LECTURER\_UNIT(lecture\_number, unit\_number)
4. UNIT(unit\_number, unit\_name)
5. STUDENT\_DETAILS(student\_no, student\_fname, student\_lname, student\_address, course\_enrolled, more\_of\_study, mentor\_number, mentor\_name)
6. ACADEMIC\_RECORD(student\_no, unit\_number, unit\_year, unit\_sem, unit\_name, grade)
7. UNIT(unit\_no, unit\_name)
8. LECTURER(lecture\_no, lecture\_fname, lecture\_lname)

**Look for relations with the same PK, then they are the same relation.**

We see that number (3), (7) and (8) have same PK; UNIT(unit\_number, unit\_name)

(3) and (8)

LECTURER\_UNIT(lecture\_number, unit\_number)

LECTURER(lecture\_no, lecture\_fname, lecture\_lname)