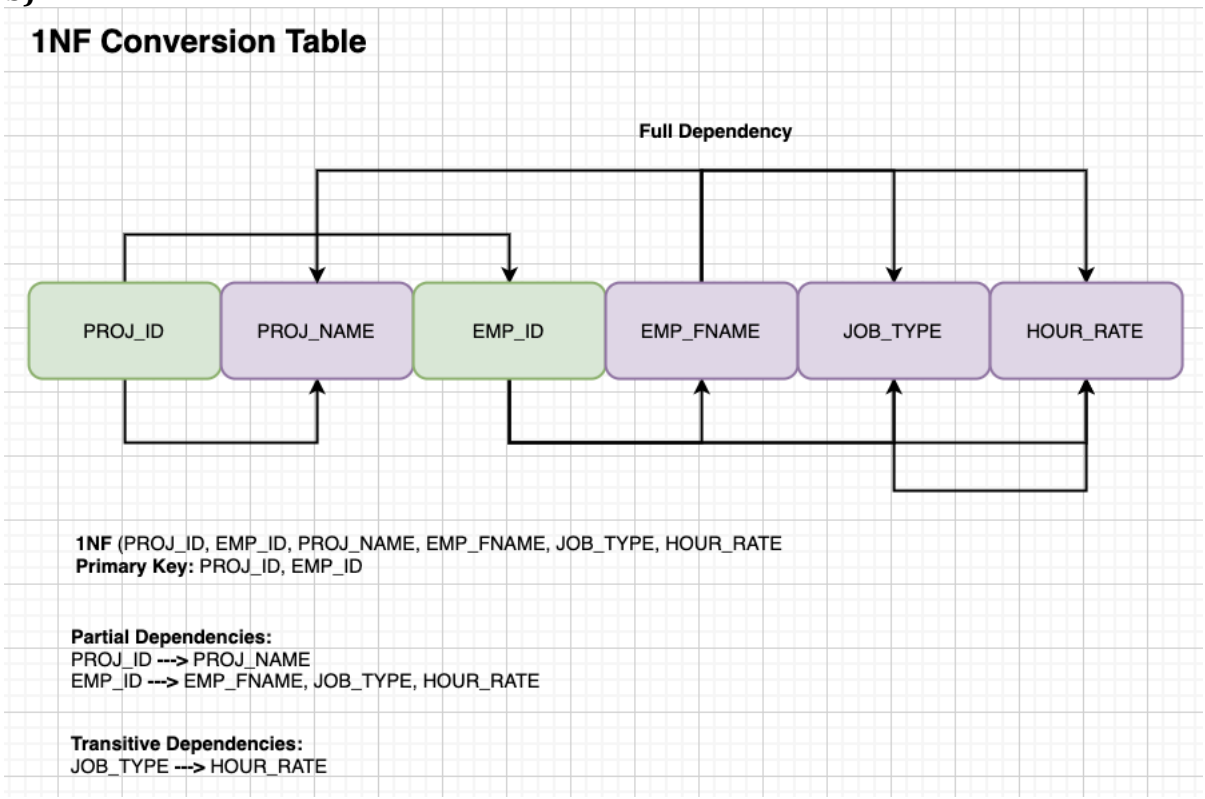
INFO151 Assignment 4, 300492683, Royan Andree Herrera Saril

1.

a) If we need to update the following: PROJ_NAME, JOB_TYPE and the HOUR_RATE then we would need to update all the records where it has been referenced. This is because if we only update one of the data then everything else in the database would be dated and therefore inconsistent.

The HOUR_RATE for the project Voyager must be an error, this is because the hourly rates for the two employees working on the project are different. Jenny works 30 and Smith works 20. It has been noted within the Assumptions and Constraints, that the hourly rate for a particular job is fixed. Which means the HOUR_RATE rows for project Voyager must be updated, either updating it to an hourly rate of 20 or an hourly rate of 30. It will need to only be updated once, where the choice must be made for which row will need to be changed.

b)

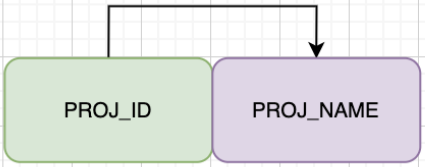


c)

2NF Conversion Table

Table Name: PROJECT

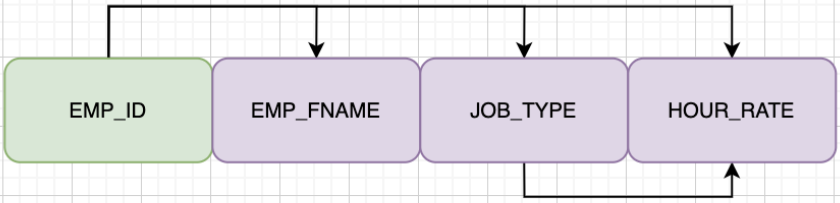
PROJECT (PROJ_ID, PROJ_NAME)



Primary Key: PROJ_ID, EMP_ID
Foreign Key: EMP_ID

Table Name: EMPLOYEE

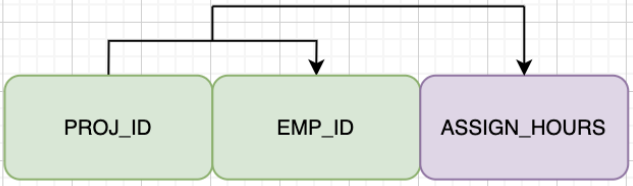
EMPLOYEE (EMP_ID, EMP_FNAME, JOB_TYPE, HOUR_RATE)



Transitive Dependencies:
JOB_CLASS ---> HOUR_RATE

Table Name: ASSIGNMENT

ASSIGNMENT (PROJ_ID, EMP_ID, ASSIGN_HOURS



3NF Conversion Table

Primary Key: PROJ_ID, EMP_ID
Foreign Key: PROJ_ID, EMP_ID, JOB_TYPE

Table Name: PROJECT

PROJECT (PROJ_ID, PROJ_NAME)

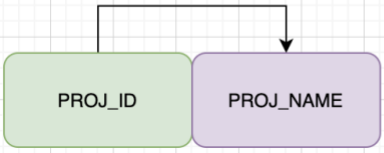


Table Name: EMPLOYEE

EMPLOYEE (EMP_ID, EMP_FNAME, JOB_TYPE)

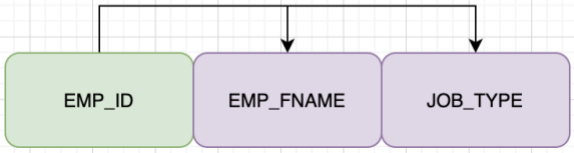


Table Name: JOB

JOB (JOB_TYPE, HOUR_RATE)

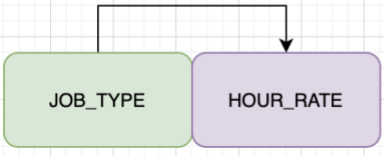
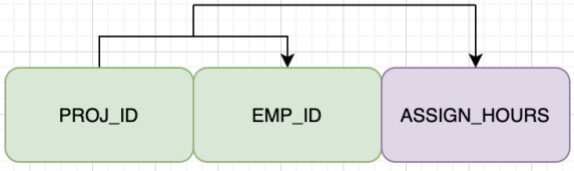


Table Name: ASSIGNMENT

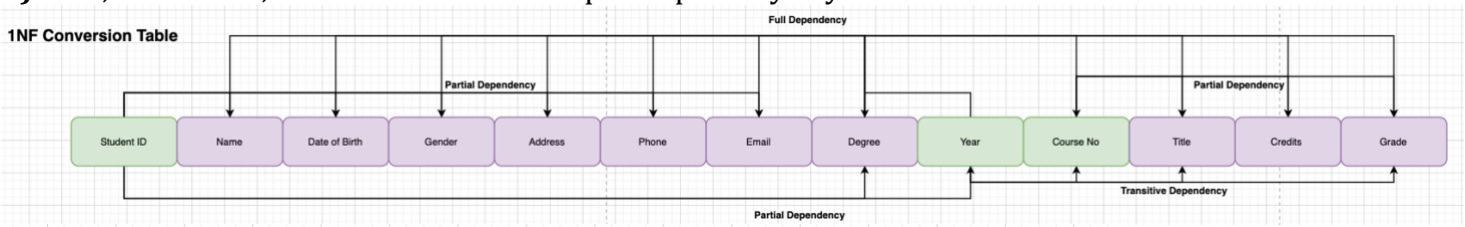
ASSIGNMENT (PROJ_ID, EMP_ID, ASSIGN_HOURS)



2.

a) Primary keys are **Student ID, Course No** and **Year**.

b) Year, Student ID, Course No are the composite primary keys.



Functional Dependency

1NF (Student ID, Name, Date of Birth, Gender, Address, Phone, Email, Degree, Year, Course No, Title, Credits, Grade)

Primary Key: Student ID, Course No, Year

Partial Dependencies:
Student ID **--->** Name, Date of Birth, Gender, Address, Phone, Email
CourseNo **--->** Title, Credits, Grade
Student ID **--->** Degree, Year

Transitive Dependency:
Year **--->** CourseNo, Title, Grade

c)

2NF Conversion Table

Table Name: STUDENT INFO

STUDENT INFO (Student ID, Name, Date of Birth, Gender, Address, Phone, Email)

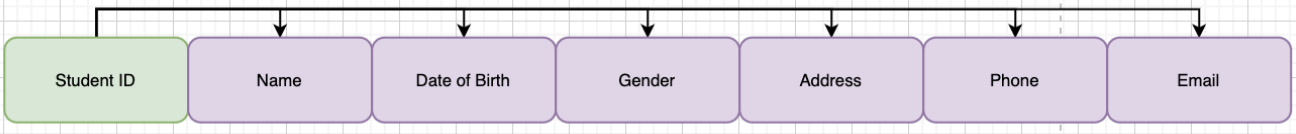
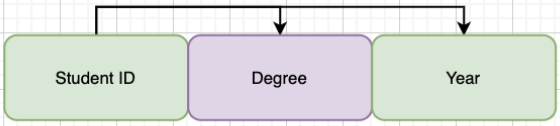


Table Name: STUDENT DEGREE

STUDENT DEGREE (Student ID, Degree, Year)



Primary Key: Student ID, Year, Course No
Foreign Key: Year, Course No

Transitive Dependency:
Year **--->** CourseNo, Title, Credits

Table Name: DEGREE

DEGREE (Year, Course No, Degree)

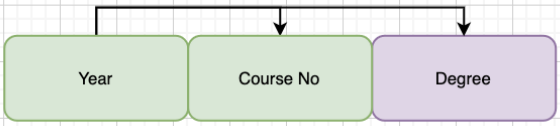
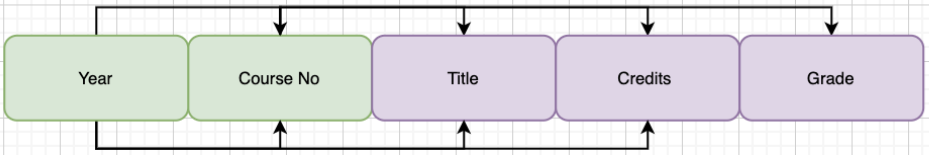
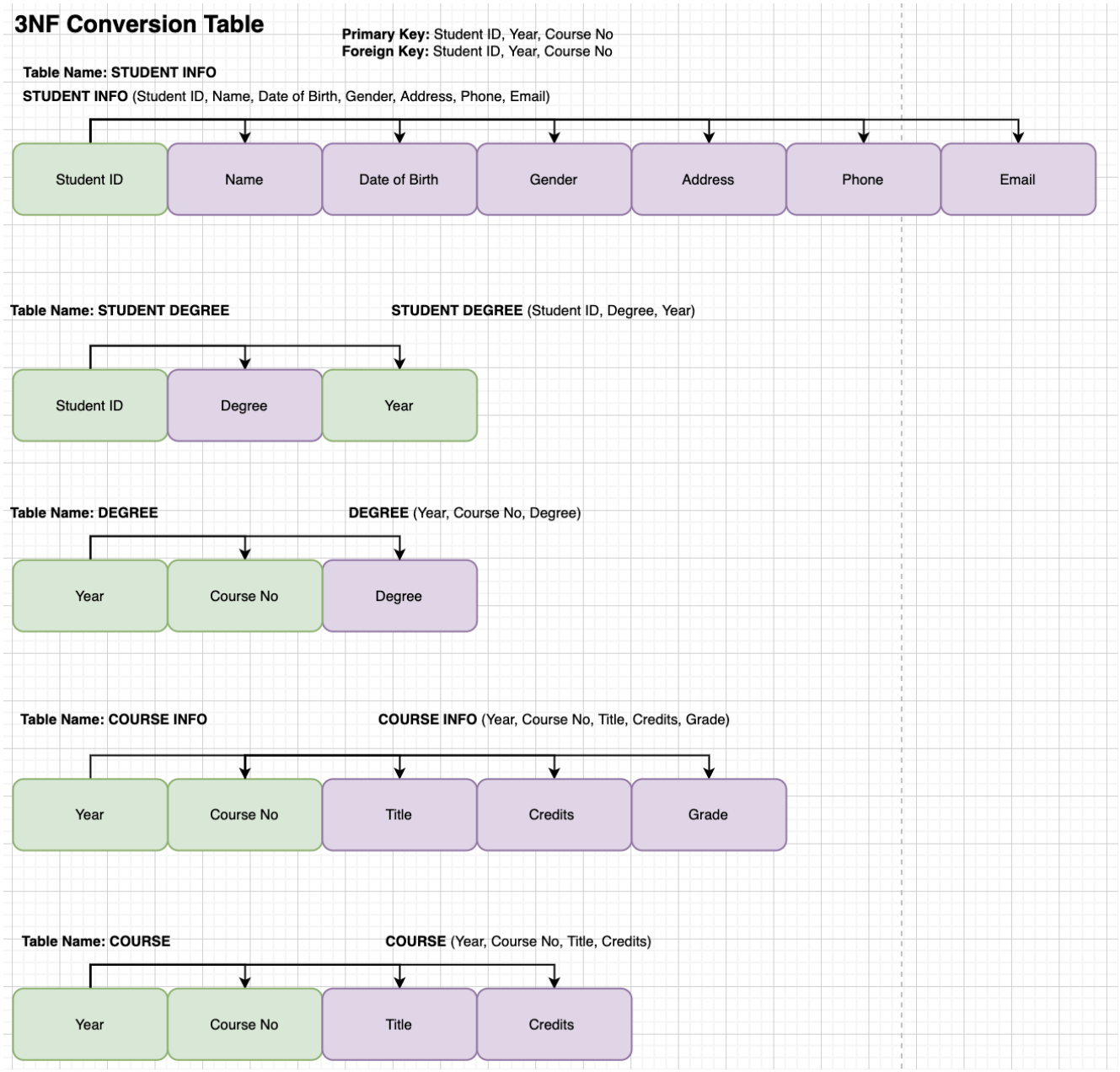


Table Name: COURSE INFO

COURSE INFO (Year, Course No, Title, Credits, Grade)





d)

STUDENT INFO

```
CREATE TABLE "Student Info"(
                            'Student ID' TEXT,
                            'Name' TEXT,
                            'Date of Birth' date,
                            'Gender' TEXT,
                            'Address' TEXT,
                            'Phone' INTEGER,
                            'Email' TEXT,
                            PRIMARY KEY ('Student Id')
                            );
```

STUDENT DEGREE

```
CREATE TABLE "Student Degree"(
                            'Student ID' TEXT,
                            'Degree' TEXT,
                            'Year' INTEGER,
                            PRIMARY KEY ('Student ID', 'Year')
                            FOREIGN KEY ('Student ID') REFERENCES 'Student Info'('Student ID')
                            );
```

DEGREE

```
CREATE TABLE "Degree"(
                        'Year' INTEGER,
                        'Course No' TEXT,
                        'Degree' TEXT
                        );
```

GRADES (changed from 'COURSE INFO')

```
CREATE TABLE "Grades"(
                        'Student ID' TEXT,
                        'Year' INTEGER,
                        'Course No' TEXT,
                        'Title' TEXT,
                        'Credits' INTEGER,
                        'Grades' TEXT,
                        PRIMARY KEY ('Student ID', 'Year', 'Course No'),
                        FOREIGN KEY ('Student ID') REFERENCES 'Student Info'('Student ID')
                        FOREIGN KEY ('Year', 'Course No') REFERENCES 'Course'('Year', 'Course No')
                        );
```

COURSE

```
CREATE TABLE "Course"(
                        'Year' INTEGER,
                        'Course No' TEXT,
                        'Title' TEXT,
                        'Credits' INTEGER
                        );
```

e)

STUDENT INFO

```
INSERT INTO "Student Info"('Student ID', 'Name', 'Date of Birth', 'Gender', 'Address', 'Phone', 'Email')
VALUES('MI47007', 'James Boon', '7/07/1977', 'Male', '10 Downing Street, Wellington', '22007007', 'JB007@gmail.com');
```

Student ID	Name	Date of Birth	Gender	Address	Phone	Email
Filter	Filter	Filter	Filter	Filter	Filter	Filter
MI47007	James Boon	7/07/1977	Male	10 Downing Street, Wellington	22007007	JB007@gmail.com

STUDENT DEGREE

```
INSERT INTO "Student Degree"('Student ID', 'Degree', 'Year')
VALUES('MI47007', 'Bachelor of Commerce', '2020');
INSERT INTO "Student Degree"('Student ID', 'Degree', 'Year')
VALUES('MI47007', 'Bachelor of Technology', '2019');
```

Student ID	Degree	Year
Filter	Filter	Filter
MI47007	Bachelor of Commerce	2020
MI47007	Bachelor of Technology	2019

DEGREE

```
INSERT INTO "Degree"('Year', 'Course No', 'Degree')
VALUES('2020', 'INFO151', 'Bachelor of Commerce');
INSERT INTO "Degree"('Year', 'Course No', 'Degree')
VALUES('2020', 'ECON130', 'Bachelor of Commerce');
INSERT INTO "Degree"('Year', 'Course No', 'Degree')
VALUES('2020', 'FCOM111', 'Bachelor of Commerce');
INSERT INTO "Degree"('Year', 'Course No', 'Degree')
VALUES('2020', 'INFO101', 'Bachelor of Commerce');

INSERT INTO "Degree"('Year', 'Course No', 'Degree')
VALUES('2019', 'INFO141', 'Bachelor of Technology');
INSERT INTO "Degree"('Year', 'Course No', 'Degree')
VALUES('2019', 'INFO151', 'Bachelor of Technology');
INSERT INTO "Degree"('Year', 'Course No', 'Degree')
VALUES('2019', 'MGMT101', 'Bachelor of Technology');
INSERT INTO "Degree"('Year', 'Course No', 'Degree')
VALUES('2019', 'QUAN102', 'Bachelor of Technology');
```

Year	Course No	Degree
Filter	Filter	Filter
2020	INFO151	Bachelor of Commerce
2020	ECON130	Bachelor of Commerce
2020	FCOM111	Bachelor of Commerce
2020	INFO101	Bachelor of Commerce
2019	INFO141	Bachelor of Technology
2019	INFO151	Bachelor of Technology
2019	MGMT101	Bachelor of Technology
2019	QUAN102	Bachelor of Technology

GRADES

```
INSERT INTO "Grades"('Student ID', 'Year', 'Course No', 'Title', 'Credits', 'Grades')
VALUES('MI47007', '2020', 'INFO151', 'Databases', '15', 'A+');
INSERT INTO "Grades"('Student ID', 'Year', 'Course No', 'Title', 'Credits', 'Grades')
VALUES('MI47007', '2020', 'ECON130', 'Microeconomic Principles', '15', 'A+');
INSERT INTO "Grades"('Student ID', 'Year', 'Course No', 'Title', 'Credits', 'Grades')
VALUES('MI47007', '2020', 'FCOM111', 'Government, Law and Business', '15', 'B+');
INSERT INTO "Grades"('Student ID', 'Year', 'Course No', 'Title', 'Credits', 'Grades')
VALUES('MI47007', '2020', 'INFO101', 'Foundations of Info Systems', '15', 'C');

INSERT INTO "Grades"('Student ID', 'Year', 'Course No', 'Title', 'Credits', 'Grades')
VALUES('MI47007', '2019', 'INFO141', 'Systems Analysis', '15', 'C');
INSERT INTO "Grades"('Student ID', 'Year', 'Course No', 'Title', 'Credits', 'Grades')
VALUES('MI47007', '2019', 'INFO151', 'Databases & SQL', '15', 'WD');
INSERT INTO "Grades"('Student ID', 'Year', 'Course No', 'Title', 'Credits', 'Grades')
VALUES('MI47007', '2019', 'MGMT101', 'Introduction to Management', '15', 'B+');
INSERT INTO "Grades"('Student ID', 'Year', 'Course No', 'Title', 'Credits', 'Grades')
VALUES('MI47007', '2019', 'QUAN102', 'Statisitcs for Business', '15', 'A+');
```

Student ID	Year	Course No	Title	Credits	Grades
Filter	Filter	Filter	Filter	Filter	Filter
MI47007	2020	INFO151	Databases	15	A+
MI47007	2020	ECON130	Microeconomic Principles	15	A+
MI47007	2020	FCOM111	Government, Law and Business	15	B+
MI47007	2020	INFO101	Foundations of Info Systems	15	C
MI47007	2019	INFO141	Systems Analysis	15	C
MI47007	2019	INFO151	Databases & SQL	15	WD
MI47007	2019	MGMT101	Introduction to Management	15	B+
MI47007	2019	QUAN102	Statisitcs for Business	15	A+

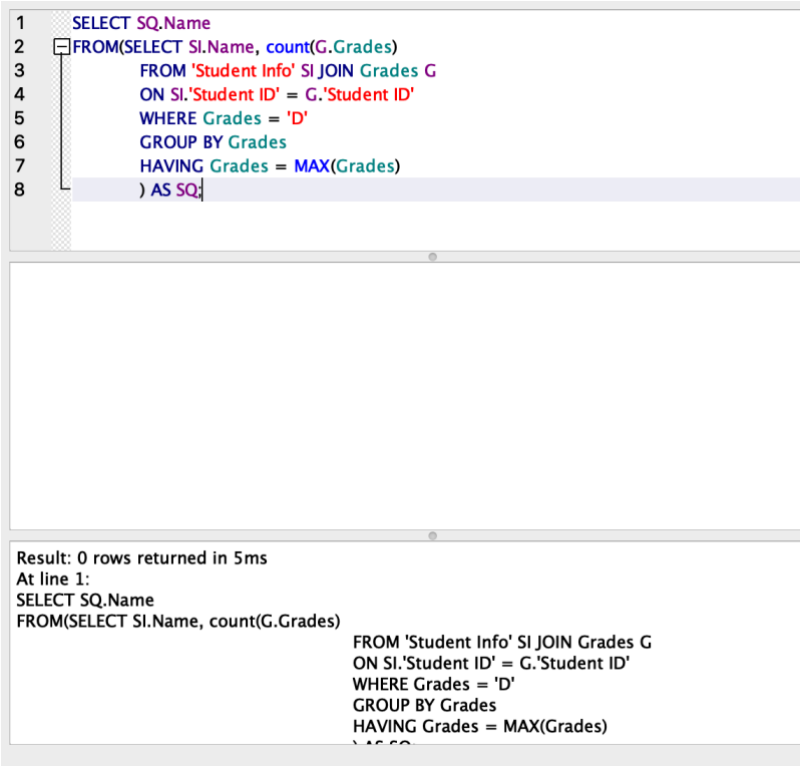
COURSE

```
INSERT INTO "Course"('Year', 'Course No', 'Title', 'Credits')
VALUES('2020', 'INFO151', 'Databases', '15');
INSERT INTO "Course"('Year', 'Course No', 'Title', 'Credits')
VALUES('2020', 'ECON130', 'Microeconomic Principles', '15');
INSERT INTO "Course"('Year', 'Course No', 'Title', 'Credits')
VALUES('2020', 'FCOM111', 'Government, Law and Business', '15');
INSERT INTO "Course"('Year', 'Course No', 'Title', 'Credits')
VALUES('2020', 'INFO101', 'Foundations of Info Systems', '15');

INSERT INTO "Course"('Year', 'Course No', 'Title', 'Credits')
VALUES('2019', 'INFO141', 'Systems Analysis', '15');
INSERT INTO "Course"('Year', 'Course No', 'Title', 'Credits')
VALUES('2019', 'INFO151', 'Databases & SQL', '15');
INSERT INTO "Course"('Year', 'Course No', 'Title', 'Credits')
VALUES('2019', 'MGMT101', 'Introduction to Management', '15');
INSERT INTO "Course"('Year', 'Course No', 'Title', 'Credits')
VALUES('2019', 'QUAN102', 'Statistics for Business', '15');
```

Year	Course No	Title	Credits
Filter	Filter	Filter	Filter
2020	INFO151	Databases	15
2020	ECON130	Microeconomic Principles	15
2020	FCOM111	Government, Law and Business	15
2020	INFO101	Foundations of Info Systems	15
2019	INFO141	Systems Analysis	15
2019	INFO151	Databases & SQL	15
2019	MGMT101	Introduction to Management	15
2019	QUAN102	Statistics for Business	15

f) Does NOT show a student who has the maximum number of D grades in the table.



3.

a) Considering the delete anomaly, if the given PatientNo was to be deleted then all the rows appointed to it would be removed as well. This means that we would have no information of the patient and the appointment and the dentist assigned.

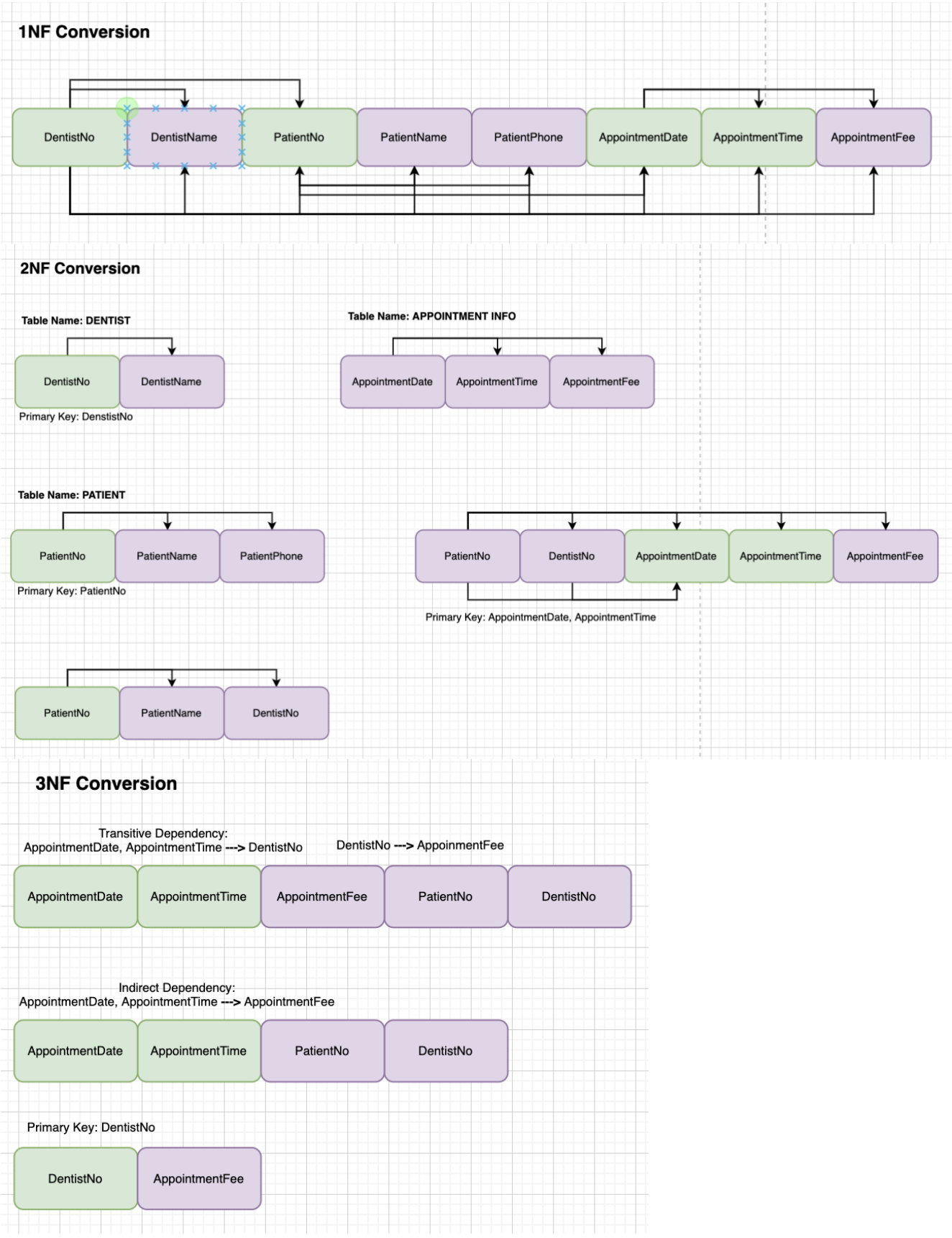
The insertion anomaly may occur when new appointments have been added, however if the patient is not immediately assigned to the PatientNo or PatientName then the patient could not be entered into the database at all. Suppose if a new dentist were also added into the database, and there are no appointments yet with this dentist then the other fields will be set to null; these fields would be the patient and appointment information.

An update anomaly would occur if an appointment would be booked twice or more at the same time, this would mean that the table would need to be update a certain number of times. Another example would be if it we were to update a patient's phone number then the update must be done for all the rows. If the update is missed for any rows, then there will be incomplete/inconsistent data.

To avoid update or delete anomalies, a take into fixing this problem would be to modify the tables into smaller tables. This would enable it so that at least one table is overlapped with another to avoid update and deletion anomalies.

b) The **DentistNo** and the **PatientNo** would be the key attributes on the table. The entity relationship between Dentist and Patient is 1:1, this is because if you take into account the appointment date and time this means that a dentist can only see ONE patient at a specific date and time, and vice versa.

c)



d) If the patient can only see a particular dentist then there would be no need for the DentistNo in the 3NF tables. This is fine because one patient will only consult with one dentist.