Normalisation Task & Data Build Task

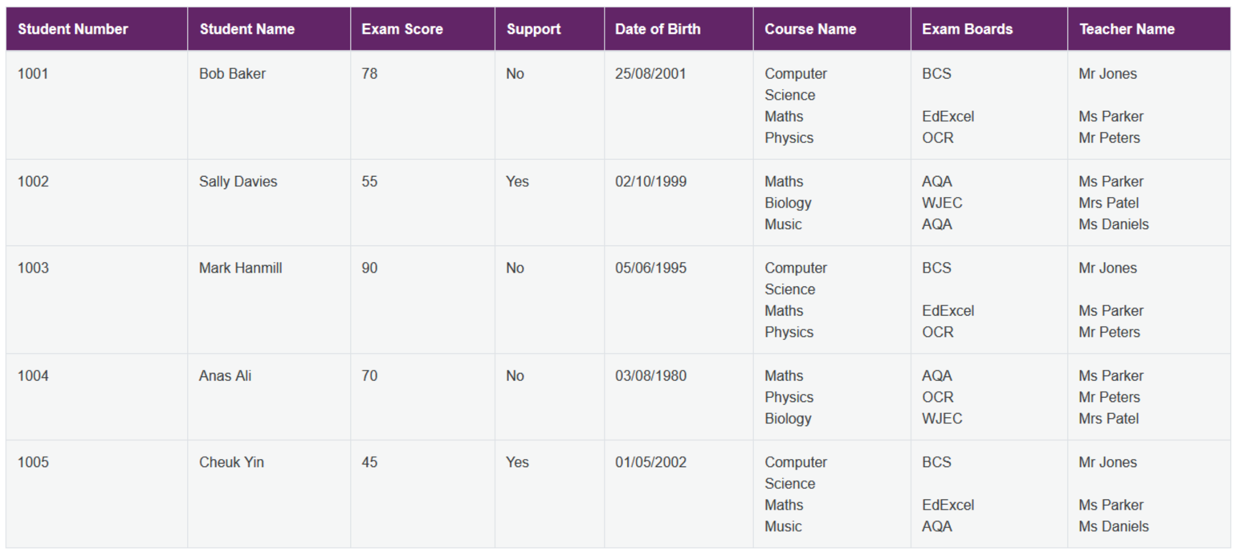
Contents

Normalisation Task

Database normalisation can be described as the process of organising data in a database or table, following a set of standard rules to reduce redundancy, speed up query processing and reduce required disk space. We can also think about database normalisation as a way to organise tables in an anticipated manner for the database users, such as, for example, having only contact details in a contact details table and not invoice number (Microsoft, 2023). The rules applied are called normal forms (NF) and is generally acceptable for data in a database to be organised till the 3rd Normal Form. More rules beyond the 3rd Normal Form do exist, such as the Boyce-Cood Normal Form or, as otherwise called, the 3.5 Normal Form, but are "rarely used outside the academic context" (Rouse, 2014).

In this exercise, we are provided with an initial data table in an un-normalised form. The task is to normalise the table down to the 3rd Normal Form by showing and explaining each subsequent step (1st Normal Form, 2nd Normal Form, 3rd Normal Form).

Table



Student Number	Student Name	Exam Score	Support	Date of Birth	Course Name	Exam Boards	Teacher Name
1001	Bob Baker	78	No	25/08/2001	Computer Science Maths Physics	BCS EdExcel OCR	Mr Jones Ms Parker Mr Peters
1002	Sally Davies	55	Yes	02/10/1999	Maths Biology Music	AQA WJEC AQA	Ms Parker Mrs Patel Ms Daniels
1003	Mark Hanmill	90	No	05/06/1995	Computer Science Maths Physics	BCS EdExcel OCR	Mr Jones Ms Parker Mr Peters
1004	Anas Ali	70	No	03/08/1980	Maths Physics Biology	AQA OCR WJEC	Ms Parker Mr Peters Mrs Patel
1005	Cheuk Yin	45	Yes	01/05/2002	Computer Science Maths Music	BCS EdExcel AQA	Mr Jones Ms Parker Ms Daniels

The table headers look as per the below table and is in 0NF (Un-normalised):

Student Number	Student Name	Exam Score	Support	Date of Birth	Course Name	Exam Boards	Teacher Name
1001	Bob Baker	78	No	25/08/2001	Computer Science Maths Physics	BCS EdExcel OCR	Mr Jones Ms Parker Mr Peters
1002	Sally Davies	55	Yes	02/10/1999	Maths Biology Music	AQA WJEC AQA	Mr Parker Mr Patel Ms Daniels

1003	Mark Hanmill	90	No	05/06/1995	Computer Science Maths Physics	BCS EdExcel OCR	Mr Jones Ms Parker Mr Peters
1004	Anas Ali	70	No	03/08/1990	Maths Physics Biology	AQA OCR WJEC	Ms Parker Mr Peters Mrs Patel
1005	Cheuk Yin	45	Yes	01/05/2022	Computer Science Maths Music	BCS EdExcel AQA	Mr Jones Ms Parker Ms Daniels

1NF

A table is considered to be in 1NF, if it (Hillyer, 2005):

1. Contains only atomic/single values in each and every cell (e.g. a cell cannot have the date of birth and date of school graduation at the same time or the address and the phone number at the same time)
2. The data type/domain attribute needs to be of the same type for each column (e.g. a cell cannot have date values and integer or text values at the same time)
3. Each column name should be unique and not be repeated.

The provided table presents the following issue, which is violating the constraint of atomic values:

- Columns 'Course Name', 'Exam Boards', and 'Teacher Name' are not atomic as they contain multiple items. For this exercise, names, we will be considered atomic values as 'full names'.

None of the other two constraints are violated.

We can also notice that duplicate values in each cell are horizontally distributed (a duplicate value in the Course Name column corresponds to a value in the Exam Board column and the Teacher Name column)

In order to solve the issues, we split our table and reformat it by duplicating each row and keeping only the atomic values. In cells where there are no atomic values, we keep the same data:

Student Number	Student Name	Exam Score	Support	Date of Birth	Course Name	Exam Boards	Teacher Name
1001	Bob Baker	78	No	25/08/2001	Computer Science	BCS	Mr Jones
1001	Bob Baker	78	No	25/08/2001	Maths	EdExcel	Ms Parker
1001	Bob Baker	78	No	25/08/2001	Physics	OCR	Mr Peters
1002	Sally Davies	55	Yes	02/10/1999	Maths	AQA	Mr Parker
1002	Sally Davies	55	Yes	02/10/1999	Biology	WJEC	Mr Patel
1002	Sally Davies	55	Yes	02/10/1999	Music	AQA	Ms Daniels
1003	Mark Hanmill	90	No	05/06/1995	Computer Science	BCS	Mr Jones
1003	Mark Hanmill	90	No	05/06/1995	Maths	EdExcel	Ms Parker
1003	Mark Hanmill	90	No	05/06/1995	Physics	OCR	Mr Peters
1004	Anas Ali	70	No	03/08/1990	Maths	AQA	Ms Parker

1004	Anas Ali	70	No	03/08/1990	Physics	OCR	Mr Peters
1004	Anas Ali	70	No	03/08/1990	Biology	WJEC	Mrs Patel
1005	Cheuk Yin	45	Yes	01/05/2022	Computer Science	BCS	Mr Jones
1005	Cheuk Yin	45	Yes	01/05/2022	Maths	EdExcel	Ms Parker
1005	Cheuk Yin	45	Yes	01/05/2022	Music	AQA	Ms Daniels

With a composite primary key in the columns Student Number and Course Name, the above table satisfies all the requirements of the 1st Normal Form; therefore, it is in 1NF. Each observation (column) may have duplicate values, but each instance (row) is unique.

2NF

A table is considered in 2NF, if it (Hillyer, 2005):

1. Is already in 1NF
2. Has no partial dependencies, meaning that every non-candidate key column should depend on the whole identified candidate key (potential primary key)

The provided table presents the following issues violating the constraints of partial dependency:

- With identified multi-value candidate key of ‘Student Number’ and ‘Course Name’, we observe that the columns ‘Exam Boards’ and ‘Teacher Name’ depend only on a part of the candidate key which is the ‘Course Name’)

In order to solve the issue, we will need to split our table into two tables by adding a foreign key to connect the two tables:

Student table

With primary key the Student Number

Student Number	Student Name	Exam Score	Support	Date of Birth
1001	Bob Baker	78	No	25/08/2001
1002	Sally Davies	55	Yes	02/10/1999
1003	Mark Hanmill	90	No	05/06/1995
1004	Anas Ali	70	No	03/08/1990
1005	Cheuk Yin	45	Yes	01/05/2022

Course Table

With composite primary key the Course Name and Student Number

Student Number	Course Name	Exam Boards	Teacher Name
1001	Computer Science	BCS	Mr Jones
1001	Maths	EdExcel	Ms Parker
1001	Physics	OCR	Mr Peters
1002	Maths	AQA	Mr Parker

1002	Biology	WJEC	Mr Patel
1002	Music	AQA	Ms Daniels
1003	Computer Science	BCS	Mr Jones
1003	Maths	EdExcel	Ms Parker
1003	Physics	OCR	Mr Peters
1004	Maths	AQA	Ms Parker
1004	Physics	OCR	Mr Peters
1004	Biology	WJEC	Mrs Patel
1005	Computer Science	BCS	Mr Jones
1005	Maths	EdExcel	Ms Parker
1005	Music	AQA	Ms Daniels

The duplicate values have also been removed from the Student table. We can observe that the Course table has the same composite key as before, creating conflict with the 2NF requirements, as there are partial dependencies with the Teacher Name being dependant on the course name only. For this reason we split the table further:

Teacher Table

With the primary key being course name

Course Name	Teacher Name
Computer Science	Mr Jones
Maths	Ms Parker
Physics	Mr Peters
Biology	Mr Patel
Music	Ms Daniels

Course Table

With composite primary key student number and course name and foreign key the course name

Student Number	Course Name	Exam Boards
1001	Computer Science	BCS
1001	Maths	EdExcel
1001	Physics	OCR
1002	Maths	AQA
1002	Biology	WJEC
1002	Music	AQA
1003	Computer Science	BCS
1003	Maths	EdExcel
1003	Physics	OCR
1004	Maths	AQA
1004	Physics	OCR
1004	Biology	WJEC
1005	Computer Science	BCS
1005	Maths	EdExcel
1005	Music	AQA

The final tables are the below:

Teacher Table

With primary key teacher name and foreign key the course name

Course Name	Teacher Name
Computer Science	Mr Jones
Maths	Ms Parker
Physics	Mr Peters
Biology	Mr Patel
Music	Ms Daniels

Course Table

With composite primary key student number and course name

Student Number	Course Name	Exam Boards
1001	Computer Science	BCS
1001	Maths	EdExcel
1001	Physics	OCR
1002	Maths	AQA
1002	Biology	WJEC
1002	Music	AQA
1003	Computer Science	BCS
1003	Maths	EdExcel
1003	Physics	OCR
1004	Maths	AQA
1004	Physics	OCR
1004	Biology	WJEC
1005	Computer Science	BCS
1005	Maths	EdExcel
1005	Music	AQA

Student table

With primary key the student number

Student Number	Student Name	Exam Score	Support	Date of Birth
1001	Bob Baker	78	No	25/08/2001
1002	Sally Davies	55	Yes	02/10/1999
1003	Mark Hanmill	90	No	05/06/1995
1004	Anas Ali	70	No	03/08/1990
1005	Cheuk Yin	45	Yes	01/05/2022

The above tables satisfy all the requirements of the 2nd Normal Form therefore they are in 2NF.

3NF

For a table to be considered in 3NF it needs to (Hillyer, 2005):

1) Be already in 2NF
2) Have no transitive dependencies, meaning that every non key column should not depend on non-key columns.

Due to the way that 2NF was achieved, all the tables already satisfy the 3rd Normal Form, and therefore they are in 3NF.

The final tables are:

Teacher Table

With primary key teacher name and foreign key the course name

Course Name	Teacher Name
Computer Science	Mr Jones
Maths	Ms Parker
Physics	Mr Peters
Biology	Mr Patel
Music	Ms Daniels

Course Table

With composite primary key student number and course name

Student Number	Course Name	Exam Boards
1001	Computer Science	BCS
1001	Maths	EdExcel
1001	Physics	OCR
1002	Maths	AQA
1002	Biology	WJEC
1002	Music	AQA
1003	Computer Science	BCS
1003	Maths	EdExcel
1003	Physics	OCR
1004	Maths	AQA
1004	Physics	OCR
1004	Biology	WJEC
1005	Computer Science	BCS
1005	Maths	EdExcel
1005	Music	AQA

Student table

With primary key the student number

Student Number	Student Name	Exam Score	Support	Date of Birth
1001	Bob Baker	78	No	25/08/2001
1002	Sally Davies	55	Yes	02/10/1999
1003	Mark Hanmill	90	No	05/06/1995
1004	Anas Ali	70	No	03/08/1990
1005	Cheuk Yin	45	Yes	01/05/2022

References

Babar, Z. (n.d.) What is BCNF?. Available from: https://www.educative.io/answers/what-is-bcnf [Accessed 28 April 2023].

Hillyer, M. (2005) An Introduction to Database Normalization. Available from: https://users.dcc.uchile.cl/~mnmonsal/cc42a/guias/intronorm.pdf [Accessed 28 April 2023].

Microsoft. (2023) Description of the database normalization basics. Available from: https://learn.microsoft.com/en-us/office/troubleshoot/access/database-normalization-description [Accessed 28 April 2023].

Rouse, M. (2014) What Does Fourth Normal Form Mean?. Available from: https://www.techopedia.com/definition/19453/fourth-normal-form-4nf [Accessed 28 April 2023].

Data Build Task

The data build task requires to create the actual relational database based on the tables of the Normalisation Task. The database should have the proper schema created, with linked tables mapped through the foreign keys, and primary key properly recorded. The database should enforce referential integrity.

Normalisation

The tables that we start with from the Normalisation Task are the below:

Teacher Table

With primary key the Course Name

Course Name	Teacher Name
Computer Science	Mr Jones
Maths	Ms Parker
Physics	Mr Peters
Biology	Mr Patel
Music	Ms Daniels

Course Table

With composite primary key the Student Number and Course Name

Student Number	Course Name	Exam Boards
1001	Computer Science	BCS
1001	Maths	EdExcel
1001	Physics	OCR
1002	Maths	AQA
1002	Biology	WJEC
1002	Music	AQA
1003	Computer Science	BCS
1003	Maths	EdExcel
1003	Physics	OCR
1004	Maths	AQA
1004	Physics	OCR
1004	Biology	WJEC
1005	Computer Science	BCS
1005	Maths	EdExcel
1005	Music	AQA

Student table

With primary key the Student Number

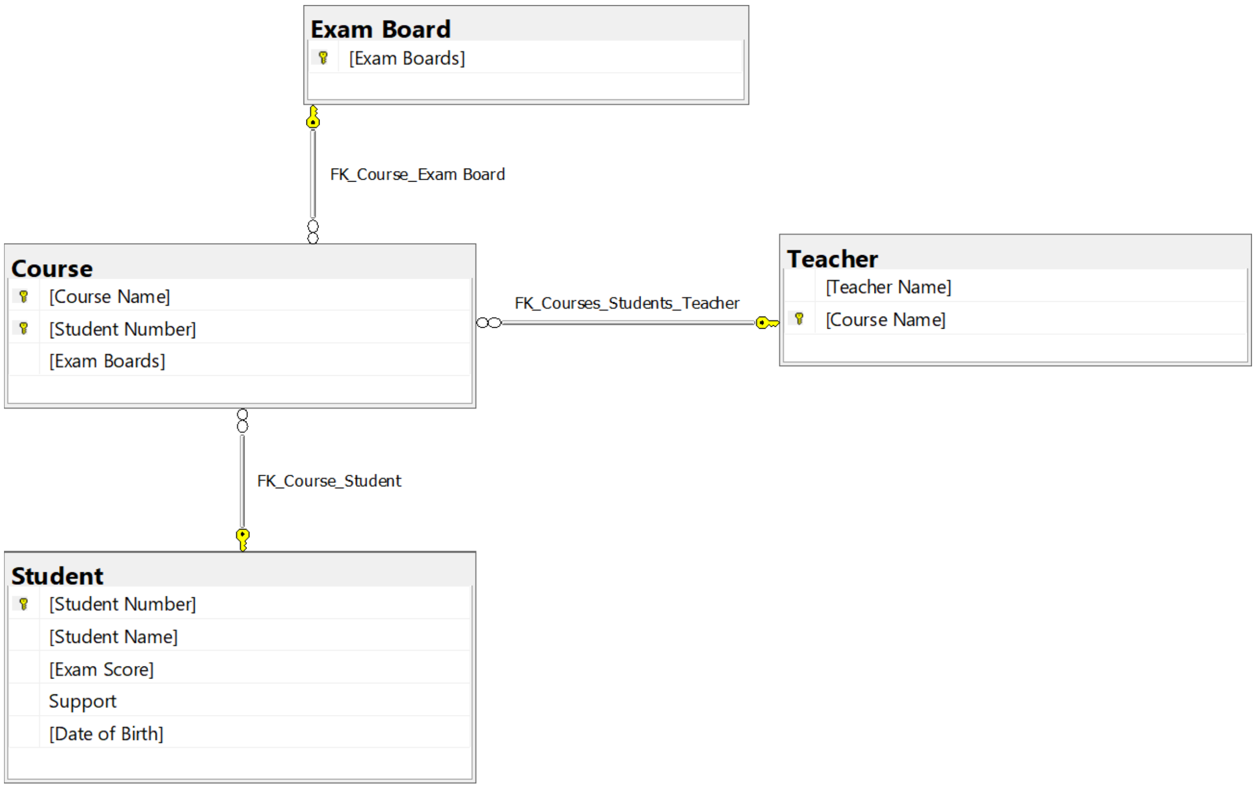
Student Number	Student Name	Exam Score	Support	Date of Birth

1001	Bob Baker	78	No	25/08/2001
1002	Sally Davies	55	Yes	02/10/1999
1003	Mark Hanmill	90	No	05/06/1995
1004	Anas Ali	70	No	03/08/1990
1005	Cheuk Yin	45	Yes	01/05/2022

Even though the database was adequately normalized we have also added an additional table name Exam Board, in order to ensure referential integrity. There was no need for an additional course table listing only the courses as the Teacher table fulfils that purpose.

Database Creation

The database tables are in the below format:



The tables are created in SSMS (SQL Server) through the below script:

```
USE [dechi]
GO
/****** Object:  Table [dbo].[Course]    Script Date: 01/05/2023 09:37:13
******/
SET ANSI_NULLS ON
```

```
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [dbo].[Course](
      [Course Name] [varchar](50) NOT NULL,
      [Student Number] [int] NOT NULL,
      [Exam Boards] [varchar](50) NOT NULL,
 CONSTRAINT [PK_Course_1] PRIMARY KEY CLUSTERED
(
      [Course Name] ASC,
      [Student Number] ASC
)WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF, IGNORE_DUP_KEY =
OFF, ALLOW_ROW_LOCKS = ON, ALLOW_PAGE_LOCKS = ON) ON [PRIMARY]
) ON [PRIMARY]
GO
/****** Object:  Table [dbo].[Exam Board]    Script Date: 01/05/2023
09:37:13 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [dbo].[Exam Board](
      [Exam Boards] [varchar](50) NOT NULL,
 CONSTRAINT [PK_Exam Board] PRIMARY KEY CLUSTERED
(
      [Exam Boards] ASC
)WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF, IGNORE_DUP_KEY =
OFF, ALLOW_ROW_LOCKS = ON, ALLOW_PAGE_LOCKS = ON) ON [PRIMARY]
) ON [PRIMARY]
GO
/****** Object:  Table [dbo].[Student]    Script Date: 01/05/2023 09:37:13
******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [dbo].[Student](
      [Student Number] [int] NOT NULL,
      [Student Name] [varchar](50) NOT NULL,
      [Exam Score] [int] NOT NULL,
      [Support] [bit] NOT NULL,
      [Date of Birth] [date] NOT NULL,
 CONSTRAINT [PK_Student] PRIMARY KEY CLUSTERED
(
      [Student Number] ASC
)WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF, IGNORE_DUP_KEY =
OFF, ALLOW_ROW_LOCKS = ON, ALLOW_PAGE_LOCKS = ON) ON [PRIMARY]
) ON [PRIMARY]
GO
/****** Object:  Table [dbo].[Teacher]    Script Date: 01/05/2023 09:37:13
******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
```

```
GO
CREATE TABLE [dbo].[Teacher](
	[Teacher Name] [varchar](50) NOT NULL,
	[Course Name] [varchar](50) NOT NULL,
 CONSTRAINT [PK_Teacher] PRIMARY KEY CLUSTERED
(
	[Course Name] ASC
)WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF, IGNORE_DUP_KEY =
OFF, ALLOW_ROW_LOCKS = ON, ALLOW_PAGE_LOCKS = ON) ON [PRIMARY]
) ON [PRIMARY]
GO
ALTER TABLE [dbo].[Course]  WITH CHECK ADD  CONSTRAINT [FK_Course_Exam
Board] FOREIGN KEY([Exam Boards])
REFERENCES [dbo].[Exam Board] ([Exam Boards])
GO
ALTER TABLE [dbo].[Course] CHECK CONSTRAINT [FK_Course_Exam Board]
GO
ALTER TABLE [dbo].[Course]  WITH CHECK ADD  CONSTRAINT [FK_Course_Student]
FOREIGN KEY([Student Number])
REFERENCES [dbo].[Student] ([Student Number])
GO
ALTER TABLE [dbo].[Course] CHECK CONSTRAINT [FK_Course_Student]
GO
ALTER TABLE [dbo].[Course]  WITH CHECK ADD  CONSTRAINT
[FK_Courses_Students_Teacher] FOREIGN KEY([Course Name])
REFERENCES [dbo].[Teacher] ([Course Name])
GO
ALTER TABLE [dbo].[Course] CHECK CONSTRAINT [FK_Courses_Students_Teacher]
GO
ALTER TABLE [dbo].[Exam Board]  WITH CHECK ADD  CONSTRAINT [FK_Exam
Board_Exam Board] FOREIGN KEY([Exam Boards])
REFERENCES [dbo].[Exam Board] ([Exam Boards])
GO
ALTER TABLE [dbo].[Exam Board] CHECK CONSTRAINT [FK_Exam Board_Exam Board]
GO
ALTER TABLE [dbo].[Teacher]  WITH CHECK ADD  CONSTRAINT
[FK_Teacher_Teacher] FOREIGN KEY([Course Name])
REFERENCES [dbo].[Teacher] ([Course Name])
GO
ALTER TABLE [dbo].[Teacher] CHECK CONSTRAINT [FK_Teacher_Teacher]
GO
ALTER TABLE [dbo].[Student]  WITH CHECK ADD  CONSTRAINT [CK_Stud_DoB]
CHECK  ((getdate()>=[Date Of Birth]))
GO
ALTER TABLE [dbo].[Student] CHECK CONSTRAINT [CK_Stud_DoB]
GO
ALTER TABLE [dbo].[Student]  WITH CHECK ADD  CONSTRAINT
[CK_Stud_ExamScore] CHECK  (([Exam Score]>=(0) AND [Exam Score]<=(100)))
GO
ALTER TABLE [dbo].[Student] CHECK CONSTRAINT [CK_Stud_ExamScore]
GO
```

Data Insertion

In order to check that our tables work as intended the data were added in the tables through the below script:

```
INSERT INTO [Student] ([Student Number], [Student Name], [Exam Score],
[Support], [Date of Birth]) VALUES ('1001', 'Bob Baker', '78', 0, '2001-
08-25');
INSERT INTO [Student] ([Student Number], [Student Name], [Exam Score],
[Support], [Date of Birth]) VALUES ('1002', 'Sally Davies', '55', -1,
'1999-10-02');
INSERT INTO [Student] ([Student Number], [Student Name], [Exam Score],
[Support], [Date of Birth]) VALUES ('1003', 'Mark Hanmill', '90', 0,
'1995-06-05');
INSERT INTO [Student] ([Student Number], [Student Name], [Exam Score],
[Support], [Date of Birth]) VALUES ('1004', 'Anas Ali', '70', 0, '1990-08-
03');
INSERT INTO [Student] ([Student Number], [Student Name], [Exam Score],
[Support], [Date of Birth]) VALUES ('1005', 'Cheuk Yin', '45', -1, '2022-
05-01');

INSERT INTO Teacher ([Course Name], [Teacher Name]) VALUES ('Computer
Science', 'Mr Jones');
INSERT INTO Teacher ([Course Name], [Teacher Name]) VALUES ('Maths', 'Ms
Parker');
INSERT INTO Teacher ([Course Name], [Teacher Name]) VALUES ('Physics', 'Mr
Peters');
INSERT INTO Teacher ([Course Name], [Teacher Name]) VALUES ('Biology', 'Mr
Patel');
INSERT INTO Teacher ([Course Name], [Teacher Name]) VALUES ('Music', 'Ms
Daniels');

INSERT INTO [Exam Board] ([Exam Boards]) VALUES ('BCS');
INSERT INTO [Exam Board] ([Exam Boards]) VALUES ('EdExcel');
INSERT INTO [Exam Board] ([Exam Boards]) VALUES ('OCR');
INSERT INTO [Exam Board] ([Exam Boards]) VALUES ('AQA');
INSERT INTO [Exam Board] ([Exam Boards]) VALUES ('WJEC');

INSERT INTO Course ([Student Number], [Course Name], [Exam Boards]) VALUES
('1001', 'Computer Science', 'BCS');
INSERT INTO Course ([Student Number], [Course Name], [Exam Boards]) VALUES
('1001', 'Maths', 'EdExcel');
INSERT INTO Course ([Student Number], [Course Name], [Exam Boards]) VALUES
('1001', 'Physics', 'OCR');
INSERT INTO Course ([Student Number], [Course Name], [Exam Boards]) VALUES
('1002', 'Maths', 'AQA');
INSERT INTO Course ([Student Number], [Course Name], [Exam Boards]) VALUES
('1002', 'Biology', 'WJEC');
INSERT INTO Course ([Student Number], [Course Name], [Exam Boards]) VALUES
('1002', 'Music', 'AQA');
INSERT INTO Course ([Student Number], [Course Name], [Exam Boards]) VALUES
('1003', 'Computer Science', 'BCS');
```

```
INSERT INTO Course ([Student Number], [Course Name], [Exam Boards]) VALUES
('1003', 'Maths', 'EdExcel');
INSERT INTO Course ([Student Number], [Course Name], [Exam Boards]) VALUES
('1003', 'Physics', 'OCR');
INSERT INTO Course ([Student Number], [Course Name], [Exam Boards]) VALUES
('1004', 'Maths', 'AQA');
INSERT INTO Course ([Student Number], [Course Name], [Exam Boards]) VALUES
('1004', 'Physics', 'OCR');
INSERT INTO Course ([Student Number], [Course Name], [Exam Boards]) VALUES
('1004', 'Biology', 'WJEC');
INSERT INTO Course ([Student Number], [Course Name], [Exam Boards]) VALUES
('1005', 'Computer Science', 'BCS');
INSERT INTO Course ([Student Number], [Course Name], [Exam Boards]) VALUES
('1005', 'Maths', 'EdExcel');
INSERT INTO Course ([Student Number], [Course Name], [Exam Boards]) VALUES
('1005', 'Music', 'AQA');
```

Referential Integrity

The following checks have been performed in the created table, in order to ensure that referential integrity is properly enforced:

- Ensuring the no duplicate teachers can be added in the Teacher table
- Ensuring that no duplicate students can be added in the Student table
- Ensuring that no duplicate exam boards can be added in the Exam Board table
- Ensuring that no duplicate combinations of student number and course name can be added in the courses table.
- Ensuring the added row in the courses table represents existing students (student table), courses (teachers table) and exam boards (exam board table)

The script used for the testing is the below:

```
-- Add same course twice under different teacher in the teacher table
INSERT INTO Teacher ([Course Name], [Teacher Name]) VALUES ('Computer
Science', 'Ms Jones');

-- Add the same student twice with different data in the student table
INSERT INTO [Student] ([Student Number], [Student Name], [Exam Score],
[Support], [Date of Birth]) VALUES ('1001', 'Bob Baker2', '78', 0, '2001-
08-12')

-- Add the same exam board twice in the exam board table
INSERT INTO [Exam Board] ([Exam Boards]) VALUES ('EdExcel');

-- Add the same combination of course name and student id the course table
INSERT INTO Course ([Student Number], [Course Name], [Exam Boards]) VALUES
('1001', 'Computer Science', 'EdExcel');
```

```
-- Attempt to add an entry in the course table while the student does not
exist
INSERT INTO Course ([Student Number], [Course Name], [Exam Boards]) VALUES
('1007', 'Computer Science', 'BCS');

-- Attempt to add an entry in the course table while the course name does
not exist
INSERT INTO Course ([Student Number], [Course Name], [Exam Boards]) VALUES
('1001', 'Computer Science2', 'BCS');

-- Attempt to add an entry in the course table while the exam board does
not exist
INSERT INTO Course ([Student Number], [Course Name], [Exam Boards]) VALUES
('1001', 'Music', 'BCSAAA');
```

Final Notes and Working Theory

The final notes and the working theory is the following:

- Additional tables created are to ensure referential integrity (e.g. exam board)
- It is good practice to create global unique identifiers (UUIDs/GUIDs) in order to ensure uniqueness of primary IDs. This would also avoid the presence of composite keys. Though these have not been created as we followed the exact columns provided in the exercise.
- Normalization may be performed in different ways and results between two database designers may differ.
- To ensure referential integrity apart from creating the primary and secondary keys we could also follow a different approach where the respective record in created automatically if it is missing on a table, during the insert queries. For example, in our design if we try to add a new student with a course under the table courses the query will return an error as the new student needs to be created in the student table first. If we create a trigger then we can ask the database to create the new student (providing some additional details), instead of returning the error.