DB

**Relational database**

**2 and 3 NORMAL FORM**

1

**OBJECTIVES FOR TODAY **

✔ Understand what is a **composite key** ✔ Understand **the 2nd normal form** ✔ Understand **the 3rd normal form**

**COMPOSITE KEY**

A **composite key** is a primary key that is **composed of more than one column** 

| **STUDENTS**  student\_id PK |
| --- |

| **SCORE**  student\_id  course\_id  score |
| --- |

1 many 1many

| **COURSE**  course\_id PK |
| --- |

**COMPOSITE KEY**

A **composite key** is a primary key that is **composed of more than one column** 

SCORE

1. Get data for **student of id 1001** 2. Get data for **course of id 2**

3. Get data for **student of id 1001 and course of id 2**

| **Student**  **ID**  1001 | **Course ID**  1 | **Score**  70 |
| --- | --- | --- |
| 1001 | 2 | 80 |
| 1002 | 1 | 100 |
| 1003 | 2 | 45 |
| 1003 | 3 | 30 |
| 1004 | 1 | 66 |

A **composite key** is a primary key that is **composed of more than one column** 

**composite key**

| **Student ID** 1001 | **Course ID**  1 | **Score**  70 |
| --- | --- | --- |
| 1001 | 2 | 80 |
| 1002 | 1 | 100 |
| 1003 | 2 | 45 |
| 1003 | 3 | 30 |
| 1004 | 1 | 66 |

Why not just create a **score id** column?

| **Score id**  1 | **Student**  **ID**  1001 | **Course ID**  1 | **Score**  70 |
| --- | --- | --- | --- |
| 2 | 1001 | 2 | 80 |
| 3 | 1002 | 1 | 100 |
| 4 | 1003 | 2 | 45 |
| 5 | 1003 | 3 | 30 |
| 6 | 1002 | 1 | 100 |

Because it can lead to the repetition of one record

| **Score id**  1 | **Student**  **ID**  1001 | **Course ID**  1 | **Score**  70 |
| --- | --- | --- | --- |
| 2 | 1001 | 2 | 80 |
| **3** | **1002** | **1** | **100** |
| 4 | 1003 | 2 | 45 |
| 5 | 1003 | 3 | 30 |
| **6** | **1002** | **1** | **88** |

For 1 student and 1 course, we should have only 1 score, and here we have 2 records! (But sometimes, this is what we want)

**ACTIVITY 1 **XX MIN

**In these following table schema, tell what column(s) compose the primary key**

HACKER RANK RESULTS

| **Member ID** 1 | **Contest ID**  1 | **Result**  40 | **Time**  5 |
| --- | --- | --- | --- |
| 2 | 3 | 50 | 10 |
| 1 | 2 | 100 | 40 |
| 2 | 2 | 50 | 60 |
| 3 | 2 | 60 | 40 |
| 3 | 1 | 70 | 10 |
| 1 | 2 | 100 | 30 |

**ACTIVITY 2 **XX MIN

**In these following table schema, tell what column(s) compose the primary key**

STUDENT

| **Student ID**  1001 | **Student Name** Lyhour | **Date of birth** XX | **Province**  XX | **Classroom ID**  2 |
| --- | --- | --- | --- | --- |
| 1002 | Kunthy | XX | XX | 1 |
| 1003 | Chum | XX | XX | 2 |
| 1004 | Sauth | XX | XX | 6 |
| 1005 | Cham | XX | XX | 4 |
| 1006 | Smey | XX | XX | 5 |
| 1007 | Ravy | XX | XX | 5 |

XX MIN **ACTIVITY 3**

**In these following table schema, tell what column(s) compose the primary key**

ENROLMENT

| **Enrolment ID** 10011002 | **Student ID**  1001 | **Course ID** 1002 |
| --- | --- | --- |
| 10021002 | 1002 | 1002 |
| 10031002 | 1003 | 1002 |
| 10031001 | 1003 | 1001 |
| 10041001 | 1004 | 1001 |
| 10051001 | 1005 | 1001 |
| 10011004 | 1001 | 1004 |
| 10021004 | 1002 | 1004 |

**Normalization** process

*DIRTY DB*

First Normal Form 1NF

RULE 1 RULE 2 RULE 3 RULE 4

Second Normal Form 2NF

**TODAY**

Third Normal Form 3NF

*CLEAN DB !*

2NF

RULE 1

The table should already be in 1st Normal Form

| **Student id**  1001 | **Name**  Lyhour | **Course**  Javascript, English |
| --- | --- | --- |
| 1002 | Thon | French, Python |
| 1003 | Kunthy | Spanish |
| 1004 | Channary | Database, Khmer |

❌

This table is **NOT NF1 :** ❌ ❌

*Each column of the table must be single values*

2NF RULE 2

All non key attributes are dependent of every columns that compose the primary key

PRIMARY KEY

STUDENT\_ID COURSE\_ID *Here :*

*Dependents on*

SCORE COMMENT

- *STUDENT\_ID and COURSE\_ID are the composite KEY* - *SCORE fully depends on STUDENT\_ID +COURSE\_ID* - *COMMENT fully depends on STUDENT\_ID +COURSE\_ID*

2NF RULE 2 Let’s have an example PRIMARY KEY

| **Student id**  1001 | **Course**  **id**  1 | **Score**  70 | **Teacher**  Clément |
| --- | --- | --- | --- |
| 1001 | 2 | 80 | Sopheak |
| 1002 | 1 | 100 | Clément |
| 1002 | 3 | 70 | Rady |
| 1003 | 2 | 85 | Sopheak |
| 1004 | 4 | 90 | Edouard |
| 1004 | 3 | 55 | Rady |

COURSE

| **Course id**  1 | **Course**  **name**  Javascript |
| --- | --- |
| 2 | English |
| 3 | Python |
| 4 | Database |

2NF RULE 2

PRIMARY KEY

| **Student id**  1001 | **Course**  **id**  1 | **Score**  70 | **Teacher**  Clément |
| --- | --- | --- | --- |
| 1001 | 2 | 80 | Sopheak |
| 1002 | 1 | 100 | Clément |
| 1002 | 3 | 70 | Rady |
| 1003 | 2 | 85 | Sopheak |
| 1004 | 4 | 90 | Edouard |
| 1004 | 3 | 55 | Rady |

Does **Score** depends of Course\_id + Student\_id ?

2NF RULE 2

PRIMARY KEY

| **Student id**  1001 | **Course**  **id**  1 | **Score**  70 | **Teacher**  Clément |
| --- | --- | --- | --- |
| 1001 | 2 | 80 | Sopheak |
| 1002 | 1 | 100 | Clément |
| 1002 | 3 | 70 | Rady |
| 1003 | 2 | 85 | Sopheak |
| 1004 | 4 | 90 | Edouard |
| 1004 | 3 | 55 | Rady |

Does **Score** depends of Course\_id + Student\_id ?

Score depends of Student id and Course id so it respects the second normal Form

2NF RULE 2

PRIMARY KEY

| **Student id**  1001 | **Course**  **id**  1 | **Score**  70 | **Teacher**  Clément |
| --- | --- | --- | --- |
| 1001 | 2 | 80 | Sopheak |
| 1002 | 1 | 100 | Clément |
| 1002 | 3 | 70 | Rady |
| 1003 | 2 | 85 | Sopheak |
| 1004 | 4 | 90 | Edouard |
| 1004 | 3 | 55 | Rady |

Does **Teacher** depends of Course\_id + Student\_id ?

2NF RULE 2

PRIMARY KEY

| **Student id**  1001 | **Course**  **id**  1 | **Score**  70 | **Teacher**  Clément |
| --- | --- | --- | --- |
| 1001 | 2 | 80 | Sopheak |
| 1002 | 1 | 100 | Clément |
| 1002 | 3 | 70 | Rady |
| 1003 | 2 | 85 | Sopheak |
| 1004 | 4 | 90 | Edouard |
| 1004 | 3 | 55 | Rady |

Does **Teacher** depends of Course\_id + Student\_id ?

Teacher only depends of course id and does not depends of Student id so it does not respect the second normal form

2NF RULE 2

**So Teacher columns need to be moved to another table**

| **Score id**  1 | **Student id**  1001 | **Course**  **id**  1 | **Score**  70 | **Teacher**  Clément |
| --- | --- | --- | --- | --- |
| 2 | 1001 | 2 | 80 | Sopheak |
| 3 | 1002 | 1 | 100 | Clément |
| 4 | 1002 | 3 | 70 | Rady |
| 5 | 1003 | 2 | 85 | Sopheak |
| 6 | 1004 | 4 | 90 | Edouard |

7 1004 3 55 Rady

COURSE ✅

| **Course id**  1 | **Course**  **name**  Javascript | **Teacher**  Clément |
| --- | --- | --- |
| 2 | English | Sopheak |
| 3 | Python | Rady |
| 4 | Database | Edouard |

**ACTIVITY 2 **XX MIN

**1. What is the primary key of the following table? 2. Does it respect the 2nd NF? Explain why**

HACKER RANK RESULTS

| **Member ID**  1 | **Contest ID**  1 | **Member**  **nickname**  The warrior | **Result**  40 | **Time**  5 |
| --- | --- | --- | --- | --- |
| 2 | 3 | The crazy | 50 | 10 |
| 1 | 2 | The warrior | 100 | 40 |
| 2 | 2 | The crazy | 50 | 60 |
| 3 | 2 | The amazing | 60 | 40 |
| 3 | 1 | The amazing | 70 | 10 |
| 1 | 2 | The warrior | 100 | 30 |

**ACTIVITY 3 **XX MIN

**1. Take the example of last activity and remove the attributes and put them in the right tables**

HACKER RANK RESULTS

| **Member ID**  1 | **Contest ID**  1 | **Member**  **nickname**  The warrior | **Result**  40 | **Time**  5 | **Programmi ng language**  PYTHON |
| --- | --- | --- | --- | --- | --- |
| 2 | 3 | The crazy | 50 | 10 | JAVASCRIPT |
| 1 | 2 | The warrior | 100 | 40 | PYTHON |
| 2 | 2 | The crazy | 50 | 60 | PYTHON |
| 3 | 2 | The amazing | 60 | 40 | PYTHON |
| 3 | 1 | The amazing | 70 | 10 | JAVASCRIPT |
| 1 | 2 | The warrior | 100 | 30 | PYTHON |

HACKER RANK CONTEST

| **Contest ID**  1 | **Contest name**  Python basics |
| --- | --- |
| 2 | Python Advanced |
| 3 | Javascript DOM |

HACKER RANK MEMBERS

| **Member ID**  1 | **Member name**  Lyhour |
| --- | --- |
| 2 | Vun |
| 3 | Sinet |

2NF

**TO SUM UP 2NF :**

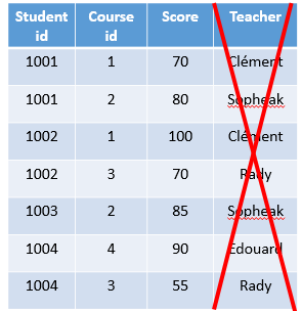
✔ Check if the table respect the 1NF

✔ What is the primary key? Is it a composite key?

✔ Check if each attribute is dependant of each column of the primary key

3NF RULE 1

The table should already be in 2nd Normal Form



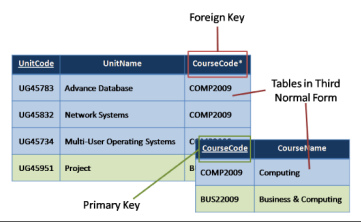
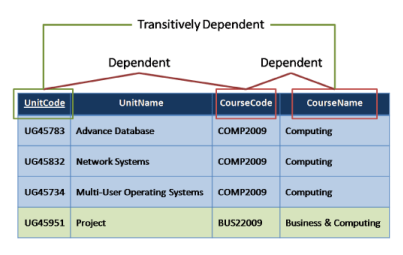
3NF RULE 2

Table has **no transitive dependencies**

3NF RULE 2

This table **is NOT 3NF**

This table **is 3NF**

****

⮚ CourseName depends on CourseCode

⮚ Each column depends on the primary key

⮚ and CourseCode is NOT the Primary key

3NF RULE 2

We have a transitive dependency if : changing a non-key column can **cause another column to also change**

| **id**  1 | **Student Name**  Lyhour | **Street**  St. 271 Toul Kok | **Town**  Phnom Penh |
| --- | --- | --- | --- |
| 2 | Thon | St. 5 Riverside | Phnom Penh |
| 3 | Kunthy | St. 2006 Resiy Koe | Phnom Penh |

• **Change street may change town**

• **Street is not a key attribute**

**ACTIVITY 4 **XX MIN

**Does the following table respect the 3rd NF?**

**COURSE**

| **COURSE ID COURSE NAME DEPARTMENT ID DEPARTMENT NAME** |
| --- |
| 1234 DATABASE 1 COMPUTING |
| 5678 C++ 1 COMPUTING |
| 7895 OPERATING SYSTEM 1 COMPUTING |

4765 OOP 2BUSINESS & COMPUTING

3NF RULE 2

How to fix it ?

✔ We have a **transitive dependency** between Street and Town :

| **id**  1 | **Student Name** Lyhour | **Street**  St. 271 Toul Kok | **Town**  Phnom Penh |
| --- | --- | --- | --- |
| 2 | Thon | St. 5 Riverside | Phnom Penh |
| 3 | Kunthy | St. 2006 Resiy Koe | Phnom Penh |

✔ We create a table address and link the 2 table with the address ID

| **id**  1 | **Student Name**  Lyhour | **Address\_id**  1 |
| --- | --- | --- |
| 2 | Thon | 2 |
| 3 | Kunthy | 3 |

| **Id**  1 | **Street**  St. 271 Toul Kok | **Town**  Phnom Penh |
| --- | --- | --- |
| 2 | St. 5 Riverside | Phnom Penh |
| 3 | St. 2006 Resiy Koe | Phnom Penh |

**ACTIVITY 4 **XX MIN

**Apply the rule 2 of the 3rd NF to the following table**

| **COURSE CODE COURSE NAME DEPARTMENT CODE DEPARTMENT NAME** |
| --- |
| 1234 DATABASE 1 COMPUTING |
| 5678 C++ 1 COMPUTING |
| 7895 OPERATING SYSTEM 1 COMPUTING |

4765 OOP 2BUSINESS& COMPUTING

3NF

**TO SUM UP 3NF : **

✔ Check if the table respect the 2NF

✔ Check if there is a dependency between 2 columns (non key columns) ✔ If so, move those columns to a new table

Summary

**NORMAL FORMS **

❖ Higher Normal Forms more closely model **relations** between **entities** ❖ This is often good, but not always!

✔ **DENORMALISED** tables can sometimes perform faster, or are easier to use, but are often less safe

❖ It can be OK for different parts of the database can have different Normal Forms

✔ e.g. we could have students/class/teacher in 3NF and books/library/students in 1NF, inside the same database