Normalization Resolved

Normalization is one of the favourite topics of interviewee. It does not matter whether you have mentioned DBMS in your resume or not .This question is going to come and the funny part is that all of us know what is normalization? What are the different types of normalization?
So when this question on being asked the interviewer who have already prepared for it start with the history of normalisation and end with the geography of normalization but when the next question for which they have not prepared i.e apply normalization in real case scenario.

Now here comes the real part of normalization and just because of not proper concepts, people end up confusing themselves. So the idea is to not only to get familiar with normalization but also how to apply it in real time scenario.

What is Normalization?

Database designed based on ER model may have some amount of inconsistency, ambiguity and redundancy. To resolve these issues some amount of refinement is required. This refinement process is called as Normalization. I know all of you are clear with the definition, let's go with:

- What is the need of normalisation?
- Why are the problems we can face if we proceed without normalisation?
- What are the advantages of normalization?

Asking question to oneself is the best way to get familiar with all the concepts.

The need of Normalization

I am going to show you one simple E-R model database.

Student Details			Course Details			Result details		
1001	Ram	11/09/1986	M4	Basic Maths	7	11/11/2004	89	A
1002	Shyam	12/08/1987	M4	Basic Maths	7	11/11/2004	78	B
1001	Ram	23/06/1987	H6		4	11/11/2004	87	A
1003	Sita	16/07/1985	C3	Basic Chemistry	11	11/11/2004	90	A
1004	Gita	24/09/1988	B3		8	11/11/2004	78	B
1002	Shyam	23/06/1988	P3	Basic Physics	13	11/11/2004	67	C
1005	Sunita	14/09/1987	P3	Basic Physics	13	11/11/2004	78	B
1003	Sita	23/10/1987	B4		5	11/11/2004	67	C
1005	Sunita	13/03/1990	H6		4	11/11/2004	56	D
1004	Gita	21/08/1987	M4	Basic Maths	7	11/11/2004	78	B

In first look the above table is looking so arranged and well in format but if we try to find out what exactly this table is saying to us, we can easily figure out the various anomalies in this table. Ok let me help you guys in finding out the same.

1. **Insert Anomaly**: We cannot insert prospective course which does not have any registered student or we cannot insert student details that is yet to register for any course.

2. **Update Anomaly**: if we want to update the course M4's name we need to do this operation three times. Similarly we may have to update student 1003's name twice if it changes.

3. **Delete Anomaly**: if we want to delete a course M4, in addition to M4 occurs details, other critical details of student also will be deleted. This kind of deletion is harmful to business. Moreover, M4 appears thrice in above table and needs to be deleted thrice.

4. **Duplicate Data**: Course M4's data is stored thrice and student 1002's data stored twice .This redundancy will increase as the number of course offerings increases.

Process of normalization:

Before getting to know the normalization techniques in detail, **let us define a few building blocks** which are used to define normal form.

1. **Determinant** : Attribute X can be defined as determinant if it uniquely defines the value Y in a given relationship or entity .To qualify as determinant attribute need NOT be a key attribute .Usually dependency of attribute is represented as X->Y ,which means attribute X decides attribute Y.

   Example: In RESULT relation, Marks attribute may decide the grade attribute .This is represented as Marks->grade and read as Marks decides Grade.

   Marks -→ Grade

   In the result relation, Marks attribute is not a key attribute .Hence it can be concluded that key attributes are determinants but not all the determinants are key attributes.

2. **Functional Dependency**: Yes functional dependency has definition but let's not care about that. Let's try to understand the concept by example. Consider the following relation :

   REPORT (Student#, Course#, CourseName, IName, Room#, Marks, Grade)

   Where:

   - Student#-Student Number
   - Course#-Course Number
   - CourseName -CourseName
   - IName- Name of the instructor who delivered the course
   - Room#-Room number which is assigned to respective instructor
   - Marks- Scored in Course Course# by student Student #
   - Grade –Obtained by student Student# in course Course #
   - Student#,Course# together (called composite attribute) defines EXACTLY ONE value of marks .This can be symbolically represented as

     Student#Course# → Marks

This type of dependency is called functional dependency. In above example Marks is functionally dependent on Student#Course#.

Other Functional dependencies in above examples are:

- Course# -> CourseName
- Course#-> IName(Assuming one course is taught by one and only one instructor)
- IName -> Room# (Assuming each instructor has his /her own and non-shared room)
- Marks ->Grade

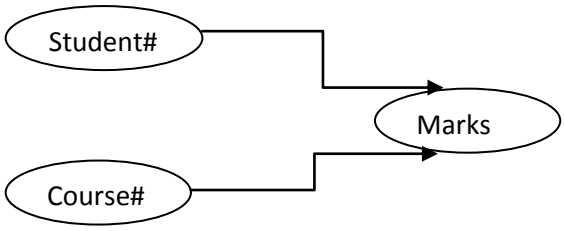
Formally we can define functional dependency as: In a given relation R, X and Y are attributes. Attribute Y is functional dependent on attribute X if each value of X determines exactly one value of Y. This is represented as :

X->Y

However X may be composite in nature.

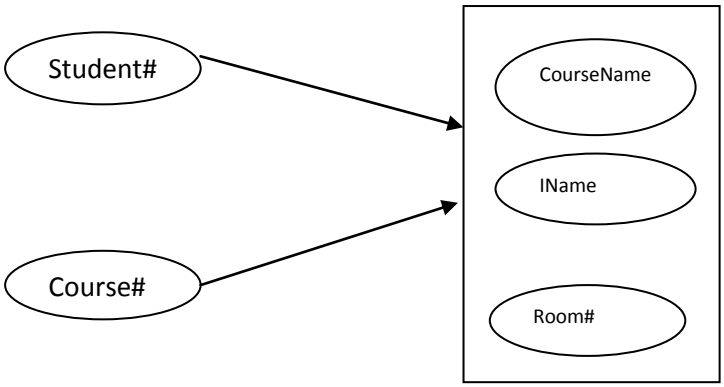
3. **Full functional dependency** : In above example Marks is fully functional dependent on *student#Course#* and not on the sub set of *Student#Course#* .This means marks cannot be determined either by student # or Course# alone .It can be determined by using Student# and Course# together. Hence Marks is fully functional dependent on student#course#.

   CourseName is not fully functionally dependent on student#course# because one of the subset course# determines the course name and Student# does not having role in deciding Course name .Hence CourseName is not fully functional dependent on student #Course#.



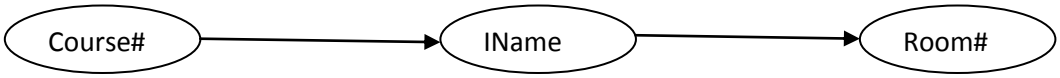
   Formal Definition of full functional dependency: In a given relation R ,X and Y are attributes. Y is fully functionally dependent on attribute X only if it is not functionally dependent on sub-set of X. However X may be composite in nature.

4. **Partial Dependency**: In the above relationship CourseName, IName,Room# are partially dependent on composite attribute Student#Course# because Course# alone can defines the coursename, IName,Room#.



Formal Definition of Partial dependency: In a given relation R, X and Y are attributes .Attribute Y is partially dependent on the attribute X only if it is dependent on subset attribute X .However X may be composite in nature.

5. **Transitive Dependency**: In above example , Room# depends on IName and in turn depends on Course# .Here Room# transitively depends on Course#.



Similarly Grade depends on Marks,in turn Marks depends on Student#Course# hence Grade Fully transitively depends on Student#Course#.

6. **Key attributes**: In a given relationship R, if the attribute X uniquely defines all other attributes, then the attribute X is a key attribute which is nothing but the candidate key.

Ex: Student#Course# together is a composite key attribute which determines all attributes in relationship REPORT (student#, Course#, CourseName, IName, Room#, Marks, Grade) uniquely. Hence Student# and Course# are key attributes.

Types of Normal Forms

1. **First Normal Form(1NF)**
   A relation R is said to be in first normal form (1NF) if and only if all the attributes of the relation R, are atomic in nature.

Table shown below Student Details, Course Details and Result Details can be further divided. Student Details attribute is divided into Student#(Student Number) , Student Name and date

of birth. Course Details is divided into Course#, Course Name,Prerequisites and duration. Similarly, Results attribute is divided into DateOfexam, Marks and Grade.

Student Details			Course Details			Result details		
1001	Ram	11/09/1986	M4	Basic Maths	7	11/11/2004	89	A
1002	Shyam	12/08/1987	M4	Basic Maths	7	11/11/2004	78	B
1001	Ram	23/06/1987	H6		4	11/11/2004	87	A
1003	Sita	16/07/1985	C3	Basic Chemistry	11	11/11/2004	90	A
1004	Gita	24/09/1988	B3		8	11/11/2004	78	B
1002	Shyam	23/06/1988	P3	Basic Physics	13	11/11/2004	67	C
1005	Sunita	14/09/1987	P3	Basic Physics	13	11/11/2004	78	B
1003	Sita	23/10/1987	B4		5	11/11/2004	67	C
1005	Sunita	13/03/1990	H6		4	11/11/2004	56	D
1004	Gita	21/08/1987	M4	Basic Maths	7	11/11/2004	78	B

2. Second Normal Form (2NF)

A relation is said to be in Second Normal Form if and only If:

- It is in the first normal form ,and
- No partial dependency exists between non-key attributes and key attributes.

Let us re-visit 1NF table structure.

- Student# is key attribute for Student ,
- Course# is key attribute for Course
- Student#Course# together form the composite key attributes for result relationship.
- Other attributes are non-key attributes.

To make this table 2NF complaint, we have to remove all the partial dependencies.

- StudentName and DateOfBirth depend only on student#.
- CourseName,PreRequisite and DurationInDays depends only on Course#
- DateOfExam depends only on Course#.

To remove this partial dependency we need to split Student_Course_Result table into four separate tables, STUDENT, COURSE, RESULT and EXAM_DATE tables as shown in figure.

STUDENT TABLE

Student #	Student Name	DateofBirth
1001	Ram	Some value
1002	Shyam	Some value
1003	Sita	Some value
1004	Geeta	Some value
1005	Sunita	Some value

COURSE TABLE

Course#	CourseName	Duration of days
C3	Bio Chemistry	3
B3	Botany	8
P3	Nuclear Physics	1
M4	Applied Mathematics	4
H6	American History	5
B4	Zoology	9

RESULT TABLE

Student#	Course#	Marks	Grade
1001	M4	89	A
1002	M4	78	B
1001	H6	87	A
1003	C3	90	A
1004	B3	78	B
1002	P3	67	C
1005	P3	78	B
1003	B4	67	C
1005	H6	56	D
1004	M4	78	B

EXAM DATE Table

Course#	DateOfExam
M4	Some value
H6	Some value
C3	Some value
B3	Some value
P3	Some value
B4	Some value

- In the first table (STUDENT), the key attribute is Student# and all other non-key attributes, StudentName and DateOfBirth are fully functionally dependant on the key attribute.
- In the Second Table (COURSE) , Course# is the key attribute and all the non-key attributes, CourseName, DurationInDays are fully functional dependant on the key attribute.
- In third table (RESULT) Student#Course# together are key attributes and all other non-key attributes, Marks and Grade are fully functional dependant on the key attributes.
- In the fourth Table (EXAM DATE) Course# is the key attribute and the non-key attribute, DateOfExam is fully functionally dependant on the key attribute.

At first look it appears like all our anomalies are taken away! Now we are storing Student 1003 and M4 record only once. We can insert prospective students and courses at our will. We will update only once if we need to change any data in STUDENT, COURSE tables. We can get rid of any course or student details by deleting just one row.

Let us analyse the RESULT Table

Student#	Course#	Marks	Grade
1001	M4	89	A
1002	M4	78	B
1001	H6	87	A
1003	C3	90	A
1004	B3	78	B
1002	P3	67	C
1005	P3	78	B
1003	B4	67	C
1005	H6	56	D
1004	M4	78	B

We already concluded that:

- All attributes are atomic in nature
- No partial dependency exists between the key attributes and non-key attributes
- RESULT table is in 2NF

Assume, at present, as per the university evaluation policy,

- Students who score more than or equal to 80 marks are awarded with “A” grade
- Students who score more than or equal to 70 marks up till 79 are awarded with “B” grade
- Students who score more than or equal to 60 marks up till 69 are awarded with “C” grade
- Students who score more than or equal to 50 marks up till 59 are awarded with “D” grade

The University management which is committed to improve the quality of education wants to change the existing grading system to a new grading system .In the present RESULT table structure,

- We don’t have an option to introduce new grades like A+ ,B- and E
- We need to do multiple updates on the existing record to bring them to new grading definition
- We will not be able to take away “D” grade if we want to.
- 2NF does not take care of all the anomalies and inconsistencies.

3. **Third Normal Form (3NF)**
   A relation R is said to be in 3NF if and only if
   - It is in 2NF
   - No transitive dependency exists between non-key attributes and key attributes.

In the above RESULT table Student# and Course# are the key attributes. All other attributes, except grade are non-partially, non – transitively dependant on key attributes. The grade attribute is dependent on "Marks "and in turn "Marks" is dependent on Student# Course#. To bring the table in 3NF we need to take off this transitive dependency.

Student#	Course#	Marks
1001	M4	89
1002	M4	78
1001	H6	87
1003	C3	90
1004	B3	78
1002	P3	67
1005	P3	78
1003	B4	67
1005	H6	56
1004	M4	78

UpperBound	LowerBound	Grade
100	95	A+
94	90	A
89	85	B+
84	80	B
79	75	B-
74	70	C
69	65	C-

After normalizing tables to 3NF, we got rid of all the anomalies and inconsistencies. Now we can add new grade systems, update the existing one and delete the unwanted ones.

Hence the Third Normal form is the most optimal normal form and 99% of the databases which require efficiency in

- INSERT
- UPDATE
- DELETE

Operations are designed in this normal form.

Hope this article will be useful to engineering students and to interviewer too!

Author: Prashant Prakash
Website: Alien Coders
Facebook Page: https://www.facebook.com/aliencoders
Twitter : http://www.twitter.com/aliencoders