COMPUTER ORGANIZATION AND ARCHITECTURE

Lecture 6

Normalization

Dr Victoria Mukami

INTRODUCTION

During the last lecture, we gained an understanding of how ERD diagrams are created. More specifically, we learnt about tables or relations, relationships, and cardinality. This lecture continues with the database lifecycle design phase, where we look at normalization. We will first review what normalization is and why it is important during database design. Next, we will review what dependencies exist before normalization. Finally, we will understand the normalization process by performing normalization on existing data.

Learning objectives

By the end of this topic, you should be able to:

1. Understand why normalization is necessary during database design
2. Review the functional dependencies before normalization
3. Determine the normalization process
4. Distinguish between INF, 2NF and 3NF

OVERVIEW

We have spent so much time on the database design phase and for good reason. One needs to be able to understand the structure of the database and make it as superior and as functional as possible. This is especially critical as a database can evolve from being simple with very few tables to being complex with many tables and even more complex relationships. Part of creating a superior database is getting rid of redundancies that may occur when designing a database. The process by which you can get rid of these redundancies is known as normalization.

WHY NORMALIZATION

Let us first define the term normalization. Normalization is the process of evaluating and correcting table structures to minimize redundancies [1]. This then reduces the likelihood of data anomalies. Normalization has three desirables levels, First Normal Form (1NF), Second Normal Form (2NF), and Third Normal Form (3NF), with 3NF being the best design. There are higher levels than 3NF, however, 3NF is considered ideal for most databases.

Once a database is designed using an ERD as we did in Lecture 5, then normalization needs to take place to try and improve the structure of the database. In most cases,

designers are requested to normalize already existing table structures. Normalization makes a database easier for a user to access and maintain the data while taking up minimal storage.

There are several objectives associated with normalization:

1. Each relation represents only a single subject. For instance, the employee table should not have customer data in it.
2. Each tuple and attribute intersection should not contain multiple values. Remember when designing ERDs and we spoke of the multi-valued attribute? Well, that is a bad design to have a record where the intersections contain multiple values.
3. Data items should not be unnecessarily stored in more than one table.
4. All non-primary key attributes on a table are only dependent on the primary key.
5. Each table has no insertion, update or deletion anomalies which then ensures the integrity and consistency of the data.

FUNCTIONAL DEPENDENCIES

One of the concepts that we need to learn is known as functional dependencies. A functional dependency is used to describe the relationship between attributes [2]. This means that the value of one or more attributes determines the value of one or more attributes. The dependency is denoted by A → B. This shows that B is functionally dependent on A. There are two main types of dependencies that exist.

Partial Dependency

This exists when there is a functional dependency where the determinant is a part of the primary key. For instance, if (A, B) → (C, D), B → C, and (A, B) is the primary key then the dependence B → C is only partially dependent as only part of the primary key from B is needed to determine the value of C and the other part would be found on A [1]. These kinds of dependencies are easy to identify.

Transitive Dependency

A transitive dependency is evident when the resulting relation is completely dependent on the primary key of the other relation. For instance, if A → B, B → C and A is the

primary key, then A → C is transitive because A is used to determine the value of C via B [1]. These types of dependencies are harder to identify.

PROCESS OF NORMALIZATION

During this lecture, we focus on the three normal forms. A summary of the normal forms is shown:

> **INF:** this data is in a tabular format and has no repeating groups with the primary key having been identified.
>
> **2NF:** this data is already in the 1NF format and has no partial dependencies
>
> **3NF:** this data is already in the 2NF format and has no transitive dependencies

Before we review the three forms, we need to understand what unnormalized data looks like.

Unnormalized Form (UNF)

This represents data that is not in tabular form. This could be data from a form or a table but contains lots of repeating groups. We will use the data in Table 1 that is unnormalized and we will move along the steps from 1NF to 3NF. Sometimes unnormalized data could be represented as form data. Think of a form that several customers will fill in their requests. The data could have data that is repeating. For instance, if the form was one where customers were making orders then there might be similar products with the same price.

Looking at Table 1, the unnormalized data has 5 main records and 8 columns. While it seems we have 5 tuples there are 20 tuples.

First Normal Form (1NF)

The first normal form involves removing all repeating groups and converting the data to tabular form. For instance, the data from Table 1 contains repeating groups where the Customer Number is used to reference several records. In the case of customer 100, there are 5 records represented. The repeating groups need to be removed as this is the first step toward reducing redundancies [1]. The second step involves identifying any primary keys within the table created.

Table 1: Unnormalized Sales Data

Customer Number	Customer Name	Employee Number	Employee Name	Employee Position	Employee Commission	Invoice Number	Invoice Amount	Invoice Date
100	Tunda Wholesalers.	500	Jane Jones	Manager	30%	10001	Ksh 45,170.00	15/01/2015
		501	Morris Mouli	Sales Agent	15%	10002	Ksh 28,998.00	04/02/2015
		502	Kevin Selabi	Customer Representative	18%	10003	Ksh 29,623.00	05/03/2015
		507	Tony Fano	Sales Agent	15%	10004	Ksh 14,075.00	13/03/2015
		503	Ann Walio	Sales Agent	15%	10005	Ksh 16,126.00	21/04/2015
101	Biashara Co.	501	Morris Mouli	Sales Agent	15%	10006	Ksh 40,353.00	15/01/2015
		503	Ann Walio	Sales Agent	15%	10007	Ksh 26,126.00	10/04/2015
		505	Rene Buha	Customer Representative	18%	10008	Ksh 34,201.00	04/02/2015
		507	Tony Fano	Sales Agent	15%	10009	Ksh 34,075.00	05/03/2015
102	Wakulima Ltd.	502	Kevin Selabi	Customer Representative	18%	10010	Ksh 42,930.00	21/04/2015
		500	Jane Jones	Manager	30%	10011	Ksh 25,170.00	15/01/2015
		501	Morris Mouli	Sales Agent	15%	10012	Ksh 48,998.00	04/02/2015
		503	Ann Walio	Sales Agent	15%	10013	Ksh 28,474.00	15/01/2015
103	Mitini Ltd.	504	Wakesho Wanga	Supervisor	25%	10014	Ksh 36,021.00	04/02/2015
		507	Tony Fano	Sales Agent	15%	10015	Ksh 17,838.00	05/03/2015
		502	Kevin Selabi	Customer Representative	18%	10016	Ksh 19,623.00	05/02/2015
		506	Lucy Kwaza	Customer Representative	18%	10017	Ksh 16,853.00	21/04/2015
104	Maziri Co.	505	Rene Buha	Customer Representative	18%	10018	Ksh 43,532.00	15/01/2015
		506	Lucy Kwaza	Customer Representative	18%	10019	Ksh 16,853.00	11/03/2015
		503	Ann Walio	Sales Agent	15%	10020	Ksh 25,474.00	15/04/2015

Step 1: Remove repeating groups [1]

This step involves converting the data into tabular form and removing the repeating groups. We do this by ensuring that cells are not merged and that each cell contains a single value. As we can see in Table 2, the repeating groups from Customer Number and Customer Name have all been assigned a customer number and customer name. Our table has been created with cells with single values. Note that we rename the column titles to more database friendly names.

Step 2: Identify the primary keys [1]

This step involves looking at the data that has been presented in Table 2 and deciding on the primary key. Just by looking at the table, CUST_NUM is not an adequate primary key. Remember, a primary key cannot be empty but more importantly needs to be unique. In our case, CUST_NUM is not unique. This shows that the primary key for our table should be a combination. Reviewing the table, we can see that a combination of a CUST_NUM, EMP_NUM and INV_NUM would make a good primary key. If we look at CUST_NO 102, EMP_NO 503, and INV_NUM 10013, it is possible to see what the records are.

Step 3: Identify all dependencies [1]

Once you identify the primary key in our case the combination of CUST_NUM and EMP_NUM then some fields are dependent on the combinational primary key. CUST_NAME, EMP_NAME, EMP_POST, EMP_COMM, INV_AMOUNT and INV_DATE are all dependent on the CUST_NUM, EMP_NUM and INV_NUM. The dependency would be shown as:

CUST_NUM, EMP_NUM, INV_NUM $\longrightarrow$ CUST_NAME, EMP_NAME, EMP_POST, EMP_COMM, INV_AMOUNT, INV_DATE

Some additional dependencies that include are shown below:

EMP_NUM $\longrightarrow$ EMP_NAME, EMP_POST, EMP_COMM

CUST_NUM $\longrightarrow$ CUST_NAME

INV_NUM $\longrightarrow$ INV_AMOUNT, INV_DATE

This means that the employee has the employee number, employee name and employee post as attributes.

Table 2: Repeating groups eliminated

CUST_NO	CUST_NAME	EMP_NO	EMP_NAME	EMP_POST	EMP_COMM	INV_NO	INV_AMOUNT	INV_DATE
100	Tunda Wholesalers	500	Jane Jones	Manager	30%	10001	Ksh 45,170.00	15/01/2015
100	Tunda Wholesalers	501	Morris Mouli	Sales Agent	15%	10002	Ksh 28,998.00	04/02/2015
100	Tunda Wholesalers	502	Kevin Selabi	Customer Representative	18%	10003	Ksh 29,623.00	05/03/2015
100	Tunda Wholesalers	507	Tony Fano	Sales Agent	15%	10004	Ksh 14,075.00	13/03/2015
100	Tunda Wholesalers	503	Ann Walio	Sales Agent	15%	10005	Ksh 16,126.00	21/04/2015
101	Biashara Co.	501	Morris Mouli	Sales Agent	15%	10006	Ksh 40,353.00	15/01/2015
101	Biashara Co.	503	Ann Walio	Sales Agent	15%	10007	Ksh 26,126.00	10/04/2015
101	Biashara Co.	505	Rene Buha	Customer Representative	18%	10008	Ksh 34,201.00	04/02/2015
101	Biashara Co.	507	Tony Fano	Sales Agent	15%	10009	Ksh 34,075.00	05/03/2015
102	Wakulima Ltd.	502	Kevin Selabi	Customer Representative	18%	10010	Ksh 42,930.00	21/04/2015
102	Wakulima Ltd.	500	Jane Jones	Manager	30%	10011	Ksh 25,170.00	15/01/2015
102	Wakulima Ltd.	501	Morris Mouli	Sales Agent	15%	10012	Ksh 48,998.00	04/02/2015
102	Wakulima Ltd.	503	Ann Walio	Sales Agent	15%	10013	Ksh 28,474.00	15/01/2015
103	Mitini Ltd.	504	Wakesho Wanga	Supervisor	25%	10014	Ksh 36,021.00	04/02/2015
103	Mitini Ltd.	507	Tony Fano	Sales Agent	15%	10015	Ksh 17,838.00	05/03/2015
103	Mitini Ltd.	502	Kevin Selabi	Customer Representative	18%	10016	Ksh 19,623.00	05/02/2015
103	Mitini Ltd.	506	Lucy Kwaza	Customer Representative	18%	10017	Ksh 16,853.00	21/04/2015
104	Maziri Co.	505	Rene Buha	Customer Representative	18%	10018	Ksh 43,532.00	15/01/2015
104	Maziri Co.	506	Lucy Kwaza	Customer Representative	18%	10019	Ksh 16,853.00	11/03/2015
104	Maziri Co.	503	Ann Walio	Sales Agent	15%	10020	Ksh 25,474.00	15/04/2015

We have a transitive dependency since the Employee Commission is dependent on the Employee Post. The dependency is shown below:

EMP_POST ⟶ EMP_COMM

By knowing the Invoice number then automatically you can tell the invoice amount and invoice date. By working on the dependencies, we can easily create a dependency diagram as shown in Figure 1. Figure 1 shows all the dependencies as seen in Table 1.

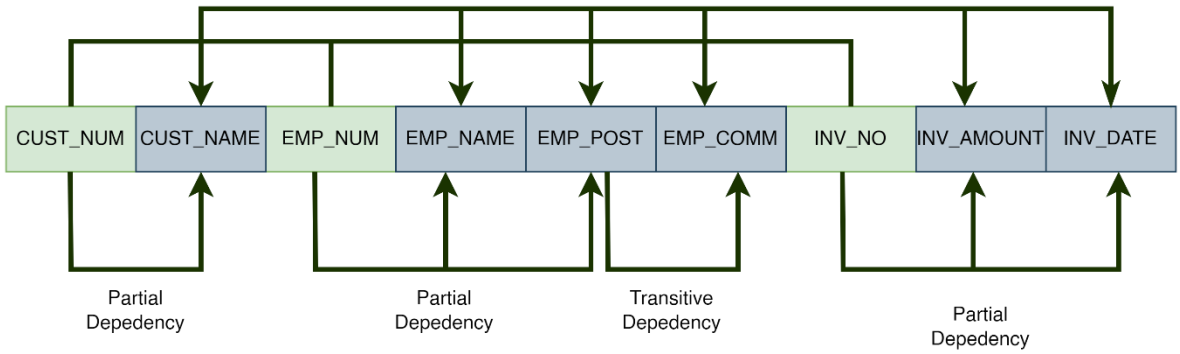

Figure 1: INF Dependency Diagram

The CUST_NUM as shown in green and all others in green are the primary keys. The arrows at the top show the dependencies as per the primary key combination. Partial and transitive dependencies are represented at the bottom. All the dependencies depend on primary keys.

Second Normal Form (2NF)

2NF happens when the table does not have a single primary key. In our case Table, 2 has several primary keys. 2NF aims to eliminate the partial dependencies by making new tables while reassigning the corresponding dependent attributes.

Step 1: Make new tables [1]

This step involves getting rid of the partial dependencies by creating new tables. In our case, we have three partial dependencies. The dependent keys are CUST_NUM, EMP_NUM and INV_NUM.

Our new tables would be the CUSTOMER, EMPLOYEE, AND INVOICE tables.

Step 2: Reassign the dependent attributes [1]

The following would be our new tables and related attributes. The primary key is underlined. Note that we still have a transitive dependency, however, the partial dependencies are all taken care of.

CUSTOMER (**CUST_NUM**, CUST_NAME)

EMPLOYEE (**EMP_NUM**, EMP_NAME, EMP_POST)

INVOICE (**INV_NUM**, INV_AMOUNT, INV_DATE)

The newly converted table is shown below.

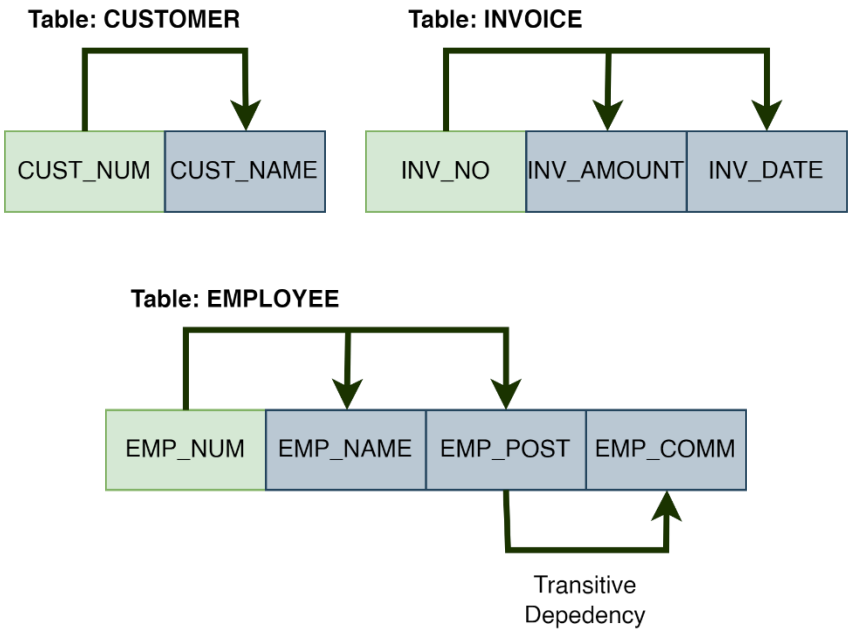

Figure 2: 2NF Converted Tables

Third Normal Form (3NF)

This step involves removing any anomalies that can be brought about by the transitive dependency [1]. Anomalies include incorrect percentages and manipulation to place higher percentages.

Step 1: Make new tables [1]

This step aims to remove transitive dependencies by creating new tables. Our transitive dependency exists on the Employee position and Employee Commission. Employee Commission is dependent on the position. We would need to come up with

a determinant that can act as a primary key for the new table. The determinant while forming the primary key, will need to remain in the original table as a foreign key. Our determinant, in this case, would be EMP_POST.

Step 2: Reassign the dependent attributes [4]

This step involves creating the new table and assigning the relevant attributes. The old EMPLOYEE table would change and is shown below.

EMP_NUM ⟶ EMP_NAME, EMP_POST

Our new table is called COMMISSION and would be represented as follows:

EMP_POST ⟶ EMP_COMM

Our completed database would look as shown in Figure 3.

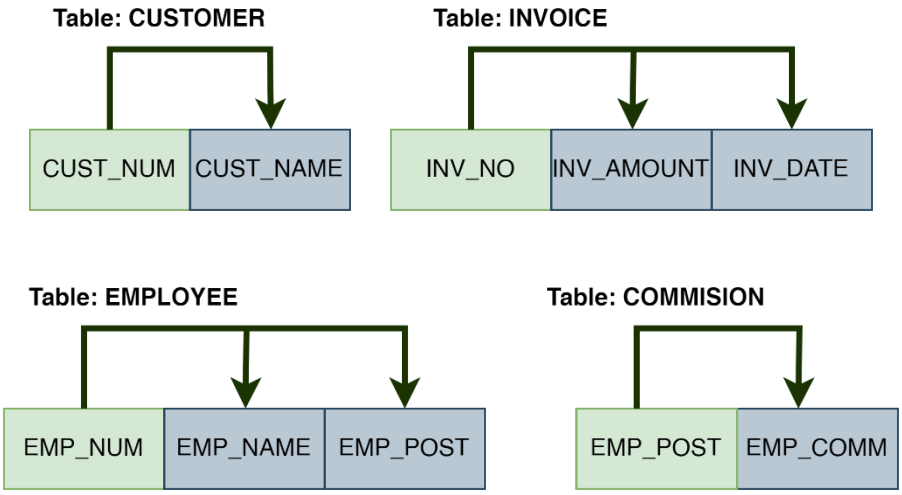

Figure 3: Final 3NF Database

This database design could be improved by [1]

1. Re-looking at the Primary Key assignments
2. Checking the naming conventions
3. Identify new attributes
4. Identify any new relationships
5. Refine the overall database

Based on the design improvement and recommendations the following would give a view of an improved database.

For instance, you could improve the design by having a table that represents the relationship between the employee and the customer and the invoice.

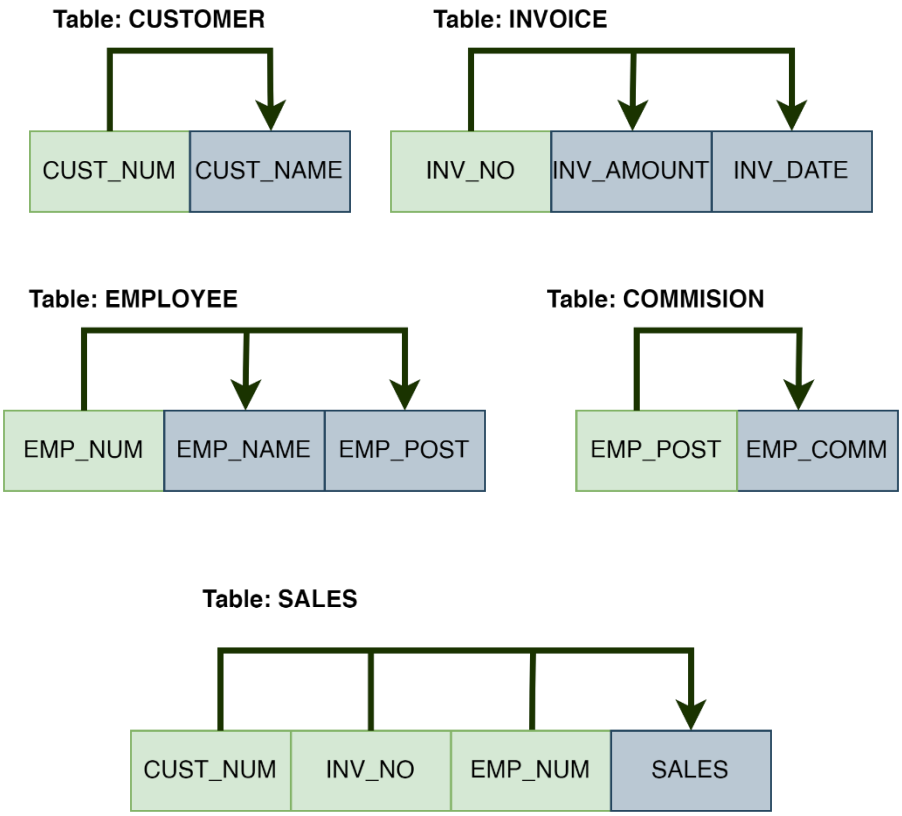

Figure 4: Final normalized tables

SUMMARY

This lecture has mainly focused on the normalization aspects of database design. We first looked at why normalization then described the types of functional dependencies. This then led us to the process of normalization, where we described INF, 2NF and 3NF. Finally, we worked on a step-by-step process of normalization from UNF to 3NF.

DISCUSSION TOPIC

During the last discussion topic, we came up with various entities and attributes. Your task now with your peers is to come up with some data that is unnormalized then go ahead and normalize the data. Compare the final database design to the initial one from lecture 5.

REFERENCES

[1] Database systems: design, implementation, and management, Coronel, C., & Morris, S, Cengage Learning, 2019.

[2] Database Systems: A Practical Approach to Design, Implementation, and Management, Connolly, T., & Begg, C., Pearson, 2015.

[3] Fundamentals of database systems, Elmasri, R., & Navathe, S. B., Pearson Education Limited, 2016.