|  |
| --- |
| Örebro University |
| Database Normalization |
| Assignment 4 - Report |

|  |
| --- |
| Duy Dinh & Özgun M  2014 - 12 - 18 |

Table of Contents

[Introduction 2](#_Toc406756974)

[Normal form rules 2](#_Toc406756975)

[TABLES 3](#_Toc406756976)

[DEPT (Department) 3](#_Toc406756977)

[EMPLOYEE 3](#_Toc406756978)

[ITEM 4](#_Toc406756979)

[PARTS 4](#_Toc406756980)

[SALE 5](#_Toc406756981)

[STORE 5](#_Toc406756982)

[SUPPLIER 6](#_Toc406756983)

[SUPPLY 6](#_Toc406756984)

[Ending words 6](#_Toc406756985)

# Introduction

In this assignment the students were given the task of analysing a demo database of a company. In the database there were different tables for each area of the company, including employees, departments, individual items, the stores, the suppliers etc.

The task was to make this demo database correct and efficient by implementing normal forms for each table and also checking for wrong information in the tables.

In this particular report the tables of the database will have their own sub-chapters where questions will be answered according to instructions.

For this assignment, the program DBVisualizer was utilized for easier displaying and manipulation of databases.

The format of the presentation for each sub-chapter is as follows:

* Possible Candidate keys.
* The chosen primary key.
* Full functional dependencies.
* Excel-table displaying small parts of each table for visualization.

## Normal form rules

Conditions for normal-forms were followed by these points.

For **1NF**:

1. There are no repeating groups
2. All data values are atomic
3. Each field has a unique name
4. The table has a primary key

For **2NF**:

1. It is in 1NF
2. All non-key attributes are dependent on ALL PARTS of the primary key

For **3NF**:

1. It is in 2NF
2. All non-key attributes are not dependent on any other non-key attributes

# TABLES

## DEPT (Department)

**Candidate keys:** {Number}

**Primary Key:** Number

**F.F.D (Full function dependency):**

* {Number -> (Name, Manager, Store, Floor)}

**Highest normal form:** BCNF

There are no dependencies between non-key attributes since the column Number is the only field that determines the other attributes, which meets the requirement of BCNF.



## EMPLOYEE

**Candidate keys:** {Number}

**Primary key:** Number

**F.F.D:**

* {(Number) -> (Name, Salary, Birthdate, Startdate)}

**Highest normal form:** BCNF

The only thing that needs to be attended to in this table is the column “Name” where it needs to be split into two different columns “Surname” and “Forename” to make it atomic. After correcting this issue, this table meets the requirement of BCNF.



## ITEM

**Candidate keys:** {Number}

**Primary key:** Number

**F.F.D:**

* {(Number) -> (Name, Dept, Price,QoH, Supplier)}

**Highest normal form:** BCNF

The table below meets the requirements for BCNF since there are no other candidate keys to depend on. All the non-key attributes are fully dependent on the primary key, which displays the item number.



## PARTS

**Candidate keys:** {PNUM}

**Primary key:** PNUM

**F.F.D:**

* {PNUM -> (PNAME, COLOR, WEIGHT, QOH)}

**Highest normal form:** BCNF

One problem with this table is that there are no ways to split it without assuming that each column can only have one set of values. This makes it hard to split name and colour into its own table since it implies that a particular part can strictly have one colour. In real world, the same part might have different colours.



## SALE

**Candidate keys:** {Item, Number-Item}

**Primary key:** Number-Item

**F.F.D:**

* {(Number, Item) -> Quantity}
* {Number -> (Date, Employee, Credit)}
* {Item -> Dept}
* {Dept -> Store}

**Highest normal form:** 1NF

This doesn’t meet the requirements for higher normal forms at first, mostly because of the several dependencies between the non-key attributes. It didn’t meet the 2NF since there were F.F.D’s to some of the other candidate keys – this violates the condition for 2NF.

This table was one of the harder ones to normalize because of the large amount of determinants. The table was split into four different tables to fulfil the requirements of 3NF and as well as BCNF:

Items Sold

Receipt

Item Area

Dept Area

**NUMBER**

**ITEM**

**QUANTITY**

**NUMBER**

**DATE**

**EMPLOYEE**

**CREDIT**

**ITEM**

**DEPT**

**DEPT**

**STORE**

100581

118

5

100581

15/01/1975

157

0

118

26

26

7

100581

120

1

100582

15/01/1975

1110

24356540

120

26

14

8

100582

26

1

100586

16/01/1975

35

54096831

26

14

49

8

## STORE

**Candidate keys:** {Number, City}

**Primary key:** Number

**F.F.D:**

* {Number -> (City, State)}
* {(City) -> (State)}

**Highest normal form:** 2NF

The reason for this is that “City” and “State” is dependent on each other. This violates the requirement for 3NF.

The field below demonstrates a complete table of BCNF:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Store |  |  | City area |  |
| **NUMBER** | **CITY** |  | **CITY** | **STATE** |
| 5 | San Francisco |  | San Francisco | Calif |
| 7 | Oakland |  | Oakland | Calif |
| 8 | El Cerrito |  | El Cerrito | Calif |

## SUPPLIER

**Candidate keys:** {Number, City}

**Primary key:** Number

**F.F.D:**

* {City -> State}

There is also a functional key in {(Name, City) -> State}

**Highest normal form:** 2NF

It doesn’t reach 3NF because there is a transitive way from “City” to “State”. Below are the tables in BCNF:



## SUPPLY

**Candidate keys:** {PNUM-JNUM}

**Primary key:** PNUM-JNUM

**F.F.D:**  There were no apparent dependencies in this table, mostly because the columns had names which were difficult to interpret and understand.

**Highest normal form:** BCNF

From looking at the fields, it was noticed that “PNUM” and “JNUM” was determining the other columns. For BCNF to be valid, each determinant needs to be a candidate key. This is true in this case.



# Ending words

There are reservations for errors, including human errors. There might be things that have been overlooked and some of the data in the tables can be incorrect in correlation to the other tables, if corrections were not implemented correctly. Overall the database tables are normalized as much as possible.