**Class: Computer Science 303: Database Management**

**Name: Myranda Brandt**

**Assignment 2**

**Identifying Functional and Transitive Dependencies**

To determine which columns are functional dependencies I need to consider what columns are completely dependent on the primary identifier of the row. For example, if we have a primary key column called one and we have another column called two, if that column one was to appear again in the table they would have the same exact value in column two in both rows. In this table I can come to the conclusion that the primary key is the EmployeeID column, as there is one row per unique EmployeeID. When I take a look at the provided table, I can see that the columns Employee Last Name, Employee First Name, Street Address, Zip Code, Department, Position and Salary are completely dependent upon the value in the Employee ID columns. If that employee ID were to be added to the table again, you would expect that all of these columns would be exactly the same, because they are all attributes of the employee identified in the Employee ID column. An employee can only have one first and last name, one current address, one current zip code, and one current salary at any given point and time. You would also imagine that an individual employee would only have one current department and position. Due to all of these columns being dependent on the employee that the row is identifying, I can come to the conclusion that these are all functional dependencies.

Next, I need to determine which columns are transitive dependencies. Transitive dependencies can be identified as columns that are dependent on another column that is not the primary identifier of the row. In this table I already determined the primary identifier is the EmployeeID column. Looking into the rows further, I can see that the City, State, and ManagerID columns are completely dependent upon other columns in the table that are not the Primary Key, and the columns they are dependent on have functional dependencies on the Primary Key column. The City and State columns are dependent upon the Zip Code column. If any rows in the table would have a certain Zip Code value displayed, those rows would also have the exact same City and State values displayed. This isn’t always accurate for the City column, as there are some Zip Codes that span multiple City boarders, but for the sake of simplicity I will assume they are all exact matches. The ManagerID columns is completely dependent on the Department column, at least in the example data given. If any row in the table were to have a certain Department value listed, they would also have the same values in the ManagerID column. In the real world this may not always be the case, but in the example data given it appears there is only one manager per department.

In conclusion, I have determined that the functional dependencies in this table are the Employee Last Name, Employee First Name, Street Address, Zip Code, Department, Position and Salary columns which are all dependent on the primary key column EmployeeID. I have also determined that the transitive dependencies in this table are the City, State, and ManagerID columns. City and State are dependent on the Zip Code column, which has a functional dependency on the EmployeeID column. ManagerID is dependent on the Department column, which has a functional dependency on the EmployeeID column.

**Identifying a Primary Key**

To identify the column which should be the primary key of the table I need to take into consideration which column holds a unique value that should not be null, and would identify a distinct row in the table. After reviewing the data provided, I can see that the Employee ID column should be the Primary Key. This column has all unique values, and the other columns in the table are dependent on what value appears here. The primary key of the table is EmployeeID.

**Explain Why Table is Not In 3NF**

For the given table to be in Third Normal Form it first has to meet some other requirements. The first requirement is that the table needs to be in first normal form. For the table to be in First Normal Form the table needs to have a primary key. In this case, the EmployeeID is the primary key. The table also needs to have single data points in each field. This table does have single values in each field, or column. There also needs to be no duplicative columns. In reviewing the table I can see each column is an independent unique attribute in the table and there are no repeated column names.

The next requirement a table needs to achieve to be in Third Normal Form is that the table needs to be in Second Normal Form. To be in Second Normal Form the table first needs to meet First Normal Form. I have already determined in the previous paragraph that the table is indeed in First Normal Form. In addition to that, all other columns need to be dependent on the primary key of the table. Looking at the sample data, all of the column here are dependent on the EmployeeID. There are no odd columns that are unrelated to the Primary Key.

Now, to be in Third Normal Form there are a few more requirements that the table needs to meet. First, the table needs to be in Second Normal Form. In the previous paragraph I explained why the table is in Second Normal Form. After that, the table needs to not have any transitive dependencies. A transitive dependency happens when a column is fully dependent on another column that is dependent on the Primary Key of the table. In this table I can see that the State and City columns are dependent on the Zip Code column, which is dependent on the primary key of the table EmployeeID. I can also see in this table that the ManagerID column is dependent on the Department column, which is dependent on the primary key of the table EmployeeID. These go against Third Normal for because the values in each of the columns I identified are determined by the column they are transitively dependent on, rather then dependent on the Primary Key. The table is not in Third Normal Form, 3NF.

The table not being in Third Normal form is not ideal due to the fact that there is redundant data in the table. Every time there is a distinct value in the Zip Code column, the same information will be repeated in the City and State columns corresponding with the zip code value. Every time there is a distinct value in the Department column, the same information will be repeated in the ManagerID column corresponding with the Department value. It is also a concern that the data integrity of the database could be in jeopardy. If an employee that is the only employee of a department leaves the company and their data is deleted from the table, the record of that department and ManagerID will also be completely deleted.

**Explain the Current Normalization Status**

I have determined that this table is not in Third Normal Form. It is not in third normal form because there are transitive dependencies present in the table. To decide what normalization status the table is in I need to start at the beginning of normalization. The table has a primary key, has a single data point in each field, and does not have duplicative columns. Due to all of these facts, the table meets the requirements for First Normal Form. The table meets First Normal Form, and all of the columns are dependent on the primary key. Due to this, the table meets Second Normal Form. The table is currently in Second Normal Form, 2NF.

**Create the Tables in MySQL**

/\*First I need to create a database to put my new tables in\*/

CREATE DATABASE cs303assignment2;

/\*Now I need to make sure I am using the new database\*/

USE cs303assignment2;

**A close-up of a computer screen

Description automatically generated**

/\*Here I will create the Employee locations table that will remove the

functional dependencies of city/state to zip code. ZipCode will be the

primary key.\*/

CREATE TABLE EmpLocations

(ZipCode VARCHAR(11) PRIMARY KEY,

City VARCHAR(100) NOT NULL,

State VARCHAR(2) NOT NULL)

;

A close up of a computer screen

Description automatically generated

/\*Here I will create the Department Heirarchy table that will remove the

functional dependencies of managerID to department. Department will be the

primary key.\*/

CREATE TABLE DepHeirarchy

(Department VARCHAR(100) PRIMARY KEY,

ManagerID INT)

;

A computer screen shot

Description automatically generated

/\*Now I will remake the employee table that will now be in third normal form,

3NF. The table will have forgien keys to the department and location tables I

made above. \*/

CREATE TABLE EmployeeMaster

(EmployeeID INT PRIMARY KEY,

EmployeeLastName VARCHAR(100),

EmployeeFirstName VARCHAR(100),

StreetAddress VARCHAR(500),

ZipCode VARCHAR(11),

Department VARCHAR(200),

Position VARCHAR(200),

Salary INT,

FOREIGN KEY (ZipCode) REFERENCES EmpLocations(ZipCode),

FOREIGN KEY (Department) REFERENCES DepHeirarchy(Department))

;

A screenshot of a computer

Description automatically generated