CMPE 232 Relational Databases Phase 2 | Final Report

Group Name: Group12

Group Members:

• Ahmet Berat Akdoğan (71569152028)

• Başar Aslan (27448161834)

• İhsan Melih Şişman (56725508390)

1) Introduction

In this final report, we indicated final versions of our E/R Diagram and Relational Schema of our Company Database. This final version consists of our changes since Phase 1 that is mainly about adding/removing entities, attributes and some updated versions regarding the type of attributes. In addition, we specified descriptions of our tables and a general explanation about Java application. We learned how to run data base operations in SQL, and we emphasize how to connect and convert to our work by using a programming language.

2) Changes that have been made since Phase 1

(+): Added, (-): Removed, (*): Changed

+ Address entity added (Attributes: Address_ID (PK), Country, City, District, Postal_Code, Address_Text).

+ Leaves entity added (Attributes: Leaves_ID (PK), Leave_Reason, Start_Date, End_Date).

+ In the Employee entity Job_Starting_Date attribute added (Data type: date).

+ In the Employee entity Status attribute added (Data type: varchar (255))

+ In the Employee entity Marriage_Status added (Data type: varchar (3)).

+ In the Employee entity AGI_Type attribute added (Data type: varchar (255)).

+ In the Employee entity AGI attribute added (Data type. numeric (6,2)).

+ In the Employee entity Total_Salary attribute added (Data type: numeric (10,2)).

+ In the Employee entity Employee_Mail attribute added (Data type: varchar (255)).

+ New attribute added to Shift entity as foreign key (Emp_ID (FK) Data type: integer) from relational schema.

+ New attribute added to Shift entity as foreign key (Dept_ID (FK) Data type: integer) from relational schema.

+ New relationship added between Shift entity and Employee entity. (1 to 1, “has”).

+ New relationship added between Address entity and Employee entity (1 to N, “has”).

+ New relationship added between Leaves entity and Employee entity (1 to 1, “rights”).

+ New attribute added to Address entity as foreign key (User_ID (FK) Data type: integer) from relational schema.

- In the Employee entity Maternity attribute has been removed.

- In the Employee entity Employee_Address attribute has been removed.

- Total participation that connects to the Employee entity (on N side) has been removed.

- In the Department entity, Shift_No attribute has been removed from relational schema.

---

* In Employee entity, the data type of Gender changed from varchar (6) to varchar (1).

* Between Shift and Department relation, the relation type changed to N to 1 from 1 to N).

* In the Shift entity, Starting_time and End_time data types changed from time to varchar (15).

* In the Device entity, Device_ID data type changed from integer to varchar (3).

* In the Employee entity, Phone_Number and Marriage_Status data types changed from integer to varchar (11) and changed from boolean to varchar (3) respectively.

* The relationship between Department and Project has been named as “includes” from “has”

* In the Project entity, the data type of Project_No has been changed from integer to varchar(3).

3) Final E/R Diagram of Our Tables

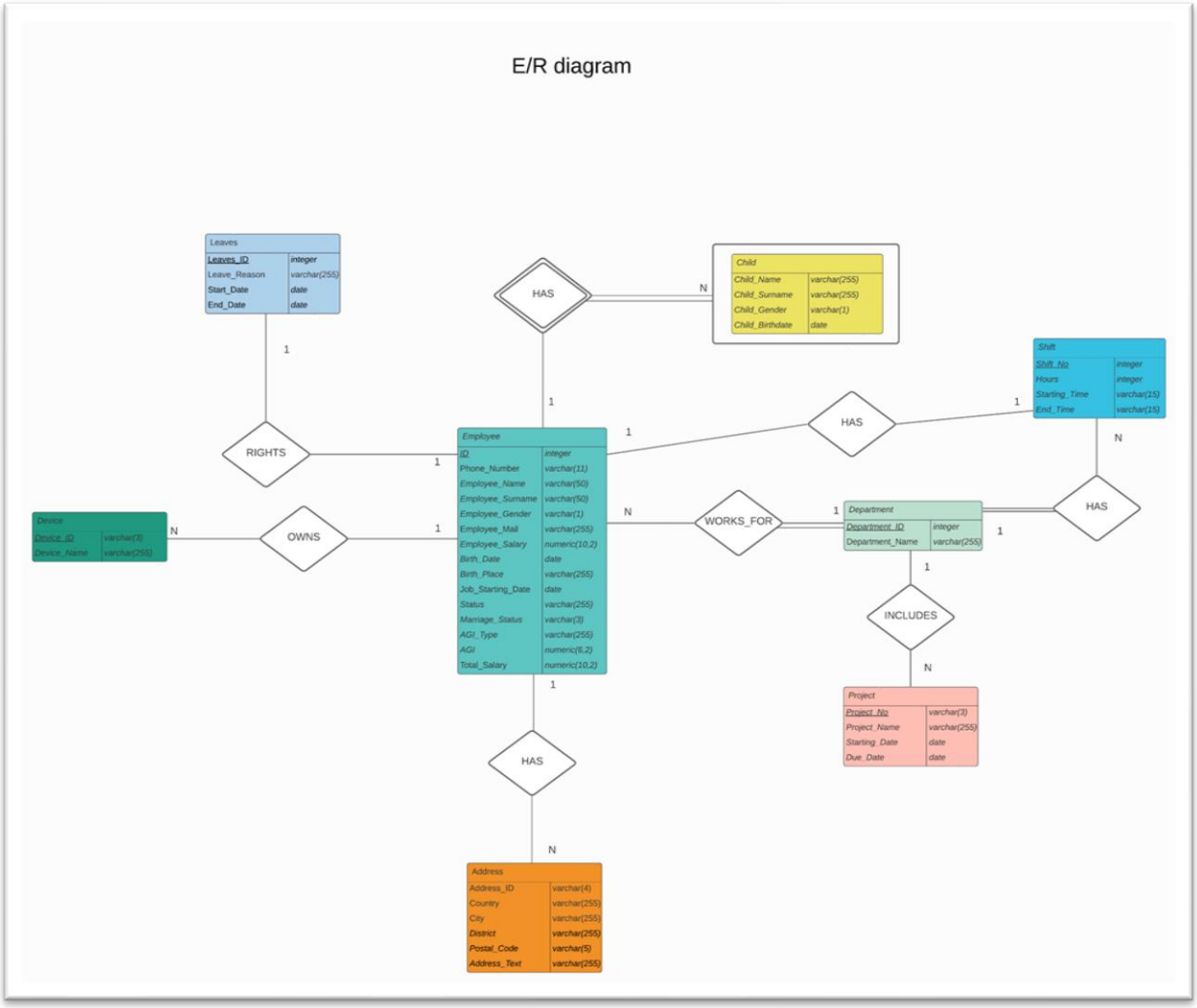

Figure 1: E/R Diagram of our Company Database

4) Final Relational Schema of Our Tables

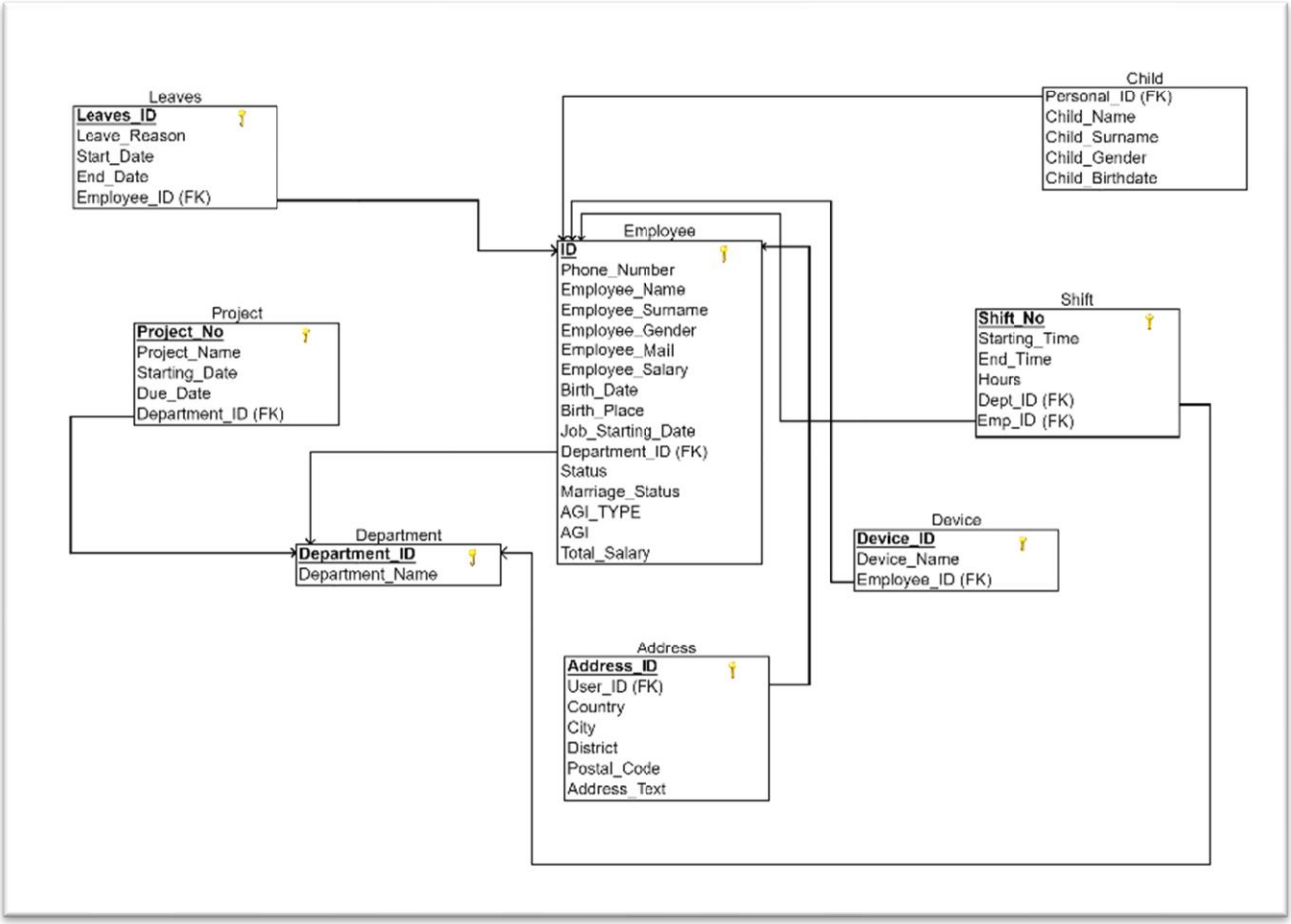

Figure 2: Final Relational Schema of our Company Database

5) Descriptions of the tables in our Database

In total, we have 8 tables that has been created. These are Employee, Child, Department, Project, Shift, Leaves, Address and Device.

Employee Table

Since many of the foreign keys connected to this entity, Employee table is our main table that we carry out most of the operations. This is our most qualified table regarding the number of attributes that holds. Our attributes which hold the general information of the employee are ID (PK), Phone_Number, Employee_Name, Employee_Surname, Employee_Gender, Employee_Salary, Birth_Date, Birth_Place, Job_Starting_Date, Status, Marriage_Status, AGI_Type, AGI, Total_Salary, Department_ID (FK). In addition to keeping the data of the employees, it aims to keep a relationship between the other tables and to obtain the data in the other table. For instance, we obtain, which employee has device thanks to the ID in the employee table.

Child Table

Secondly, we hold the data of the children of the employee in the Child table. Child table has a dependency with the Employee table. Since there is no primary key, we named this table as weak entity because every employee does not have a child. The attributes are Personal_ID (FK), Child_Name, Child_Surname, Child_Birthdate, Child_Gender. Since every child must have a parent which is an employee, we added a total participation on the Child side (on N). We connected our relationship as weak relationship since it is connected to the weak entity.

Department Table

In this department table, our aim is to hold id's and the names of department. Our primary key Department_ID has a connection with the Department_ID (FK) in the Employee entity to see which worker is working in which department. Additionally, we hold Department_Name attribute in this table.

Device Table

We created such a table because it provides electronic devices to some employees of our company. We gave the devices to the specific employees whose status and department is applicable. The attributes of this table are Device_ID (PK), Device_Name and Employee_ID (FK).

Shift Table

In this table, we arranged some shift hours due to the COVID-19 Pandemic. We aim to create numerous shift periods in order to gain flexible working hours among employees as much as possible. On that table, the attributes are Shift_No (PK), Starting_Time, End_Time, Hours, Dept_ID (FK), Emp_ID (FK). Thanks to these foreign keys, we can access Department and Employees in our Shift design.

Project Table

We created such a table because a company cannot be imagined without a project. We made the distribution of these projects according to its departments. Thus, which department was working on which project could be found more easily. The attributes are Project_No (PK), Project_Name, Starting_Date, Due_Date, Department_ID (FK).

Leaves Table

In this table our goal is to keep information of our Employee who temporarily leaves from the company. We keep a Leave_Reason attribute explains about why the employee will not come to work temporarily such as maternity, COVID-19 or some recent accident that occurred. Moreover, we specified starting and ending dates of this leaving by attributes.

Address Table

We decided to keep the address as an entity rather than an attribute because the address information of people is stored in detail in company databases and is generally not kept in an attribute. Therefore, in this entity, main information about the addresses such as city, district, postal code and address text have been stored. Thanks to the User_ID (FK), we are accessing the which employee has which address in detail.

6) Explanation of the Java application program

In this project, our main goal is to connect to our company database and to have various procedures done. We learned that adding, removing, and updating information from a database can be done in a programming language, just like we did in SQL. As a first step, we used our host, username, and password in dbeaver to connect our database by using Eclipse. To provide a console application, we used if-else statements with a scanner object to the understand user's choice. For input 1,3, and 5, our aim is listing departments, devices, and projects, respectively. We distribute update, insertion, and deletion commands respectively 2, 4, and 6. To exit from the application, user can enter the number 7. In the listing operations by help of executeQuery method, we execute the selected SQL statement with respect to our table. With getstring method, we can reach specific attributes of our table. It gets the value of the defined column in the current row. When we are updating, deleting, and inserting a value, we used setstring method to establish our operations with the correlated values. After that in each operation like inserting, deleting, and updating, we used executeupdate method to see current changes in our application. We used preparedStatement notion because it is used to execute parameterized query. For undefined data, a question mark (?) is used to describe the undefined that user will enter in the corresponding value.