Milestone 1

Database Design

COMP23311

- **Amaan Ahmad**

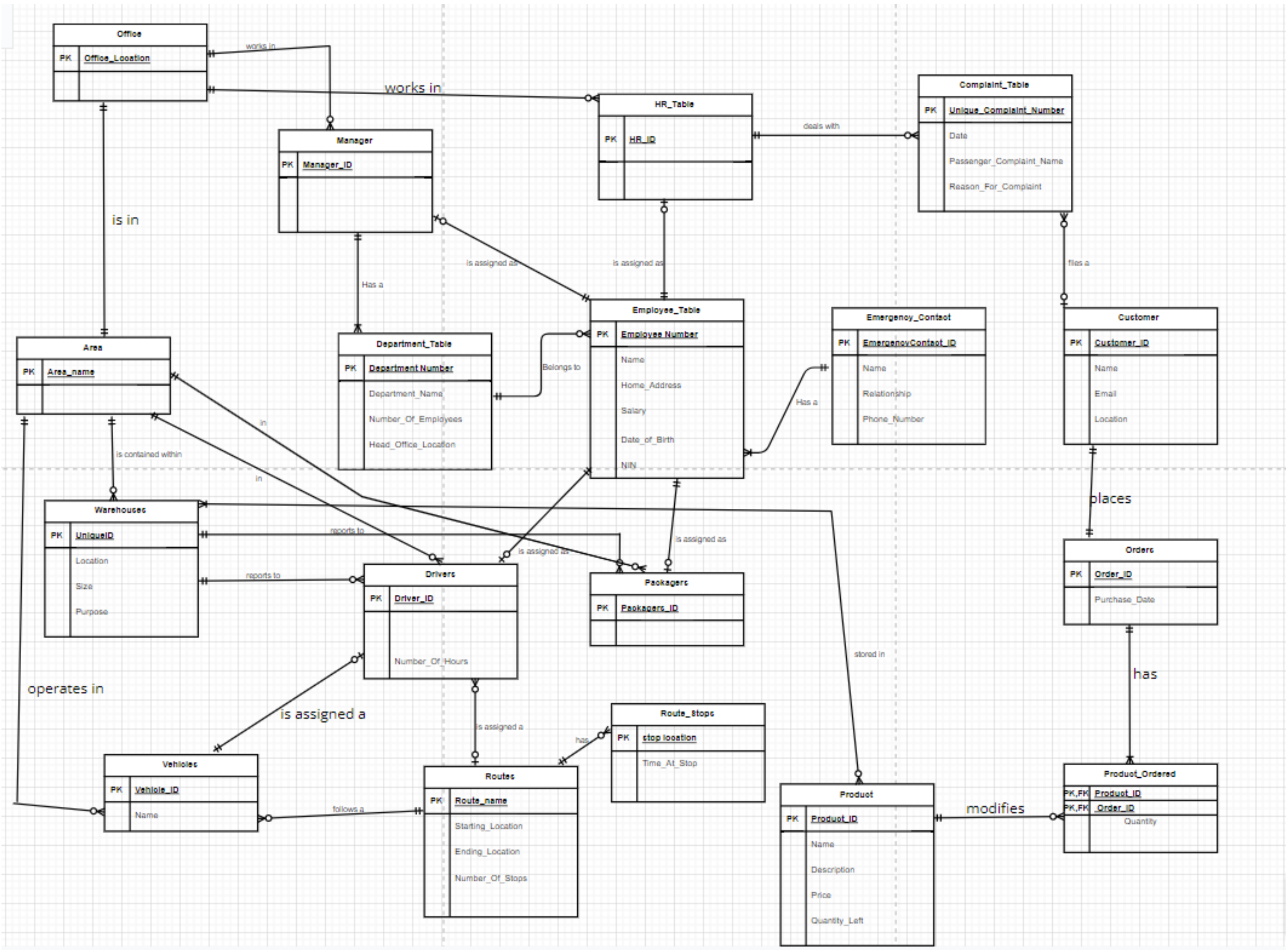
Table Of Contents:

ERD

INTRODUCTION

As the Database Engineer of the company Kilburnazon, I have created an ERD in Crow's Foot notation, that follows the Design Specification provided to us.

DIAGRAM



REPORT:

My ERD contains 18 entity tables with various attributes. I created Employee table that belongs to a Department. Each department can have multiple employees, while an employee can be in only one department. I assumed the four entities: Manager, HR, Drivers, and Packagers that are assigned as Employee has Primary Key: Manager_ID, HR_ID, Drivers_ID, and Packagers_ID respectively. These primary keys were created to avoid weak entities by using employee number. Any role, for example,a driver will be an employee but an employee may or may not be a driver(0-1 and 1-1 relation). I linked Manager with Department ,in a 1-1 and 1-many relation, as each Department has a Manager, and a manager maybe managing one or more department.

I have created a new EmergencyContact_ID as primary key for Emergency Contact. An Employee will have only one emergency contact but an emergency contact may belong to more than one employee.

As Area was linked with Warehouses, Vehicles, Office, Drivers, Packagers. I created it as an entity with PK as Area name. Warehouses\Vehicles belong to\operate in their specific area while an area may have multiple warehouses\vehicles. I have also created Office as an entity, with PK as Office_Location, which is in a specific Area(1-1 relation).

Drivers and Packagers report to the warehouse. A driver or packager will report to a single warehouse while a warehouse may have multiple drivers or packagers reporting(1-1 and 0-many relation). Manager and HR works in the Office. The relation is similar to drivers and packagers.

The Driver is assigned to only one Vehicle and Route. I did not link Routes with Area as in my ERD, Vehicle follows a Route, and it also operates in a specific Area. Therefore, Routes is linked with Area through Vehicles. A route may have multiple drivers on it (0-many and 0-1 relation). In Routes table, I recorded the starting and ending location, and the number of stops in the route. Every route has 0 or many Route Stops, which record stop location and time at stop.

I have created a primary key: Customer_ID for Customer table, along with the given attributes. Every time a Customer places a Order(1-1 relation), customer_ID will get the customer's details. I have created a separate table for Products Ordered. As it depends on the order, I gave it a PK of Product_ID and Order_ID, and an attribute Quantity. A order will have at least one product(1-1 and 1-many relation). As the product is ordered, it modifies the Product table. The product ordered have to exist in product table, while its not necessary that every item from product table is in ordered product. I have linked product table with Warehouses to keep track of items and their quantity left. A product(any quantity) can be in multiple warehouses.

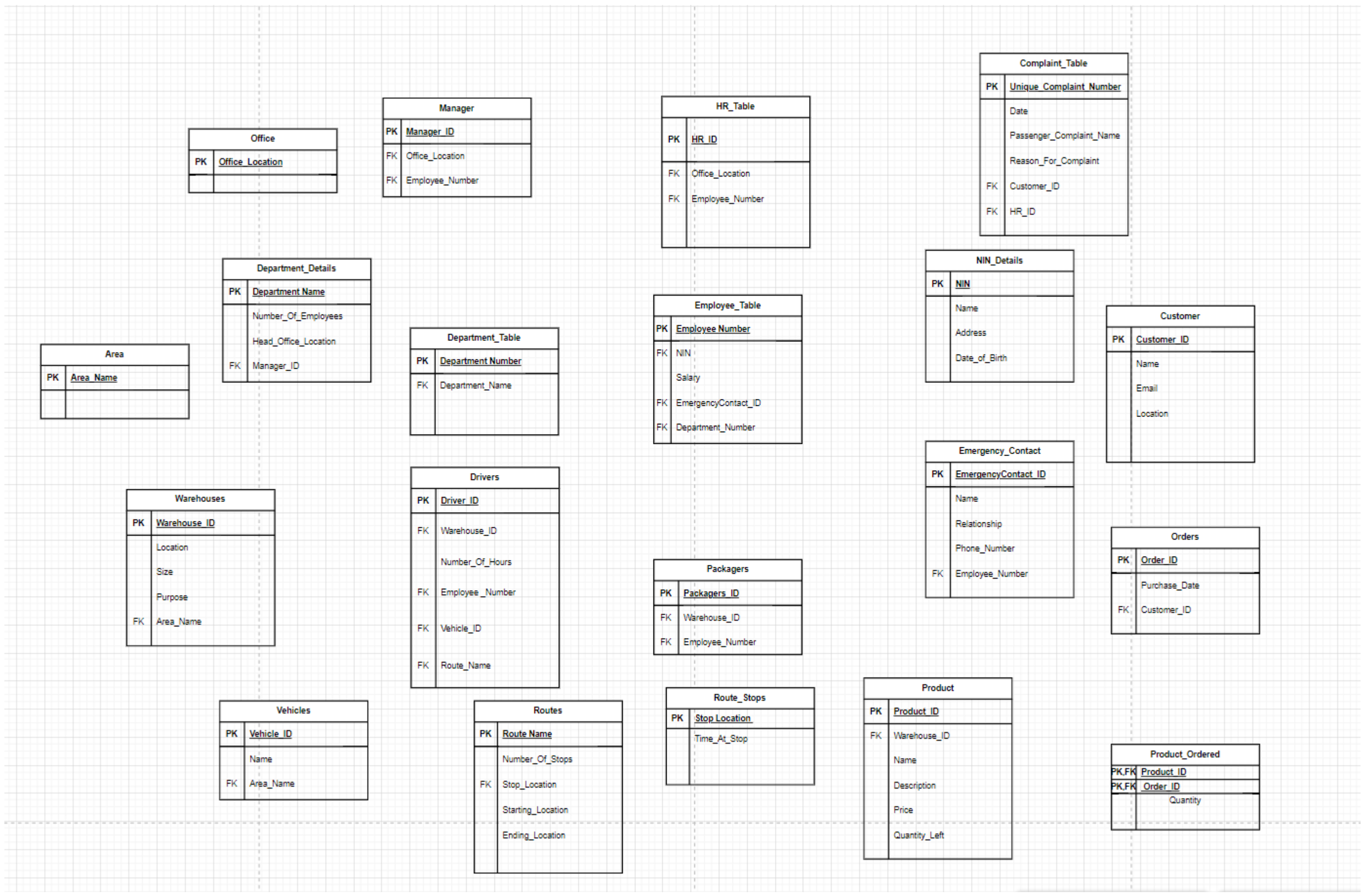
Lastly, if the costumer wants (0 to 1 relation) he/she, can file one or more Complaint(1 to many). One complaint will be dealt with by a single HR, while a HR can work on multiple(0 or many) complaint.

NORMALIZATION

INTRODUCTION

After creating ERD for the databases. Now we need to check for any redundancy in our tables to make it more organized by minimising redundancy which will be implemented making changes in our tables and making it follow 1NF,2NF, and 3NF. In order to show relations between various tables I have added foreign keys.

DIAGRAM



TEXTUAL SCHEMA:

Department_Table(Department_Number)

FK Department_Name -> Department_Details(Department_Name)

ON DELETE NO ACTION, ON UPDATE CASCADE

Department_Details(Department_Name, Number_Of_Employees, Head_Office_Location)

FK Manager_ID -> Manager(Manager_ID)

ON DELETE SET NULL, ON UPDATE CASCADE

Manager(Manager_ID)

FK Office_Location -> Office(Office_Location)

ON DELETE SET NULL, ON UPDATE CASCADE

FK Employee_Number ->Employee_Table(Employee_Number)

ON DELETE NO ACTION, ON UPDATE CASCADE

Office(Office_Location)

Employee_Table(Employee_Number, Salary)

FK NIN -> NIN_Details(NIN)

ON DELETE NO ACTION, ON UPDATE CASCADE

FK EmergencyContact_ID -> Emergency_Contact(EmergencyContact_ID)

ON DELETE SET NULL, ON UPDATE CASCADE

FK Department_Number -> Department_Table(Department_Number)

ON DELETE NO ACTION, ON UPDATE CASCADE

NIN_Details(NIN, Name, Address, Date_Of_Birth)

Emergency_Contact(EmergencyContact_ID, Name, Relationship, Phone_Number)

FK Employee_Number ->Employee_Table(Employee_Number)

ON DELETE NO ACTION, ON UPDATE CASCADE

Area(Area_Name)

HR_Table(HR_ID)

FK Office_Location -> Office(Office_Location)

ON DELETE SET NULL, ON UPDATE CASCADE

FK Employee_Number ->Employee_Table(Employee_Number)

ON DELETE NO ACTION, ON UPDATE CASCADE

Drivers(Driver_ID, Number_Of_Hours)

FK Warehouse_ID -> Warehouses(Warehouse_ID)

ON DELETE SET NULL, ON UPDATE CASCADE

FK Employee_Number ->Employee_Table(Employee_Number)

ON DELETE NO ACTION, ON UPDATE CASCADE

FK Vehicle_ID -> Vehicles(Vehicle_ID)

ON DELETE SET NULL, ON UPDATE CASCADE

FK Route_Name -> Routes(Route_Name)

ON DELETE SET NULL, ON UPDATE CASCADE

Vehicles(Vehicle_ID, Name)

FK Area_Name -> Area(Area_Name)

ON DELETE NO ACTION, ON UPDATE CASCADE

Routes(Route_Name, Number_Of_Stops, Starting_Location, Ending_Location)

FK Stop_Location -> Routes_Stops(Stop_Location)

ON DELETE NULL, ON UPDATE CASCADE

Routes_Stops(Stop_Location, Time_At_Stop)

Warehouses(Warehouse_ID , Location, Size, Purpose)

FK Area_Name -> Area(Area_Name)

ON DELETE NO ACTION, ON UPDATE CASCADE

Packagers(<u>Packagers_ID</u>)

FK Warehouse_ID -> Warehouses(Warehouse_ID)

ON DELETE SET NULL, ON UPDATE CASCADE

FK Employee_Number ->Employee_Table(Employee_Number)

ON DELETE NO ACTION, ON UPDATE CASCADE

Customer(<u>Customer_ID</u>, Name, Email, Location)

Orders(<u>Order_ID</u>, Purchase_Date)

FK Customer_ID ->Customer(Customer_ID)

ON DELETE NO ACTION, ON UPDATE CASCADE

Product(<u>Product_ID</u>, Name, Description, Price, Quantity_Left)

FK Warehouse_ID -> Warehouses(Warehouse_ID)

ON DELETE SET NULL, ON UPDATE CASCADE

Products_Ordered(<u>Product_ID, Order_ID</u>, Quantity)

FK Order_ID -> Orders(Order_ID)

ON DELETE NO ACTION, ON UPDATE CASCADE

FK Product_ID->Products(Product_ID)

ON DELETE NO ACTION, ON UPDATE CASCADE

Complaints_Table(<u>Unique_Complaint_Number.</u> Date, Passenger_Complaint_Name, Reason_For_Complaint)

FK HR_ID -> HR_Table(HR_ID)

ON DELETE NO ACTION, ON UPDATE CASCADE

FK Costumer_ID -> Customer(Costumer_ID)

ON DELETE NO ACTION, ON UPDATE CASCADE

REPORT:

1NF

To Normalize the ERD, we first have to check each table to be in 1st NF. A table will be in 1st NF if It contains no repeating groups and the values in each column are atomic. Starting from Manager, HR, Packagers: Employee_ID, Office_Location/Warehouse_ID are FKs, that have atomic values, and there are no repeating groups in the table. Employee table has salary and FKs: Department_ID, EmergencyContact_ID, NIN. All the attributes are atomic and non-repeating. Similarly, Department table and Department details satisfy 1NF requirements. Further, Office, Area, Customer, Order, Complaints, and other tables fulfil 1NF conditions. Product Ordered is an associative entity as it has composite PK (Order_ID,Product_ID), that is also FK.

The driver table(FKs: Warehouse_ID, Employee_ID, Vehicle_ID, Route_Name), has 4 FKs that link it to other entities. All these attributes have atomic values, and there are no repeating groups in the table. We could have made Route Stops' attributes in Routes but as stop location has non-atomic value(One route may have multiple stops), we made a separate table for stops for it to be in 1NF.

Hence, all tables are is 1NF.

2NF

To check for 2NF, we need to make sure the tables are in 1NF and there is no partial dependencies. If we look at every non-key attribute in Employee, Department, and Emergency Contact, its functionally dependant on the PK. Likewise, all the roles: Manager, HR, Packagers, and Drivers, have no partial dependencies. Similarly, order, customer, product, product ordered, warehouse, vehicles, and routes has non-key attributes that are functionally dependant on the primary key.

To summarize, all the tables in the ERD fulfil the requirements of 2NF.

3NF

To check for 3NF, we need to make sure the tables are in 2NF and there is no transitive dependencies. Initially, the employee table (PK: Employee Number) had name, salary, address, date of birth, and NIN as attributes. The NIN creates a transitive dependency. Let A be employee number, B be NIN, and C be Name. Then $A \rightarrow B$ and $B \rightarrow C$ (NIN is linked to the employee's details) and $A \rightarrow C$. Therefore, I created a new table called NIN Details that has NIN as primary key and Employee name, address, date of birth as attributes. NIN was then used as a foreign key in the employee table. This will avoid transitive dependency.

Similarly, the department table(PK: department_number) have department name that is unique. Department name depends on the department_number, and any other attribute (say head office location) depends on department name, and head office location depends on department_number too. Therefore, there is a transitive dependency. Hence, I created a new table named Department details, that has department name as PK, and other details of department. The department table now have only one attribute other than PK: Department name as FK.

Other tables in the ERD don't have any transitive dependency.

Hence, all the tables are now normalized as they are in 3rd Normal Form now.

RELATIONAL SCHEMA

INTRODUCTION

The content of the normalized form is expressed in text format in the relational schema below.

SCHEMA

Department_Table(<u>Department_Number</u>)

FK Department_Name -> Department_Details(Department_Name)

ON DELETE NO ACTION, ON UPDATE CASCADE

Department_Details(<u>Department_Name</u>, Number_Of_Employees, Head_Office_Location)

FK Manager_ID -> Manager(Manager_ID)

ON DELETE SET NULL, ON UPDATE CASCADE

Manager(<u>Manager_ID</u>)

FK Office_Location -> Office(Office_Location)

ON DELETE SET NULL, ON UPDATE CASCADE

FK Employee_Number ->Employee_Table(Employee_Number)

ON DELETE NO ACTION, ON UPDATE CASCADE

Office(<u>Office_Location</u>)

Employee_Table(<u>Employee_Number</u>, Salary)

FK NIN -> NIN_Details(NIN)

ON DELETE NO ACTION, ON UPDATE CASCADE

FK EmergencyContact_ID -> Emergency_Contact(EmergencyContact_ID)

ON DELETE SET NULL, ON UPDATE CASCADE

FK Department_Number -> Department_Table(Department_Number)

ON DELETE NO ACTION, ON UPDATE CASCADE

NIN_Details(NIN, Name, Address, Date_Of_Birth)

Emergency_Contact(EmergencyContact_ID, Name, Relationship, Phone_Number)

FK Employee_Number ->Employee_Table(Employee_Number)

ON DELETE NO ACTION, ON UPDATE CASCADE

Area(Area_Name)

HR_Table(HR_ID)

FK Office_Location -> Office(Office_Location)

ON DELETE SET NULL, ON UPDATE CASCADE

FK Employee_Number ->Employee_Table(Employee_Number)

ON DELETE NO ACTION, ON UPDATE CASCADE

Drivers(Driver_ID, Number_Of_Hours)

FK Warehouse_ID -> Warehouses(Warehouse_ID)

ON DELETE SET NULL, ON UPDATE CASCADE

FK Employee_Number ->Employee_Table(Employee_Number)

ON DELETE NO ACTION, ON UPDATE CASCADE

FK Vehicle_ID -> Vehicles(Vehicle_ID)

ON DELETE SET NULL, ON UPDATE CASCADE

FK Route_Name -> Routes(Route_Name)

ON DELETE SET NULL, ON UPDATE CASCADE

Vehicles(Vehicle_ID, Name)

FK Area_Name -> Area(Area_Name)

ON DELETE NO ACTION, ON UPDATE CASCADE

Routes(Route_Name, Number_Of_Stops, Starting_Location, Ending_Location)

FK Stop_Location -> Routes_Stops(Stop_Location)

ON DELETE NULL, ON UPDATE CASCADE

Routes_Stops(Stop_Location, Time_At_Stop)

Warehouses(Warehouse_ID , Location, Size, Purpose)

FK Area_Name -> Area(Area_Name)

ON DELETE NO ACTION, ON UPDATE CASCADE

Packagers(Packagers_ID)

FK Warehouse_ID -> Warehouses(Warehouse_ID)

ON DELETE SET NULL, ON UPDATE CASCADE

FK Employee_Number ->Employee_Table(Employee_Number)

ON DELETE NO ACTION, ON UPDATE CASCADE

Customer(Customer_ID, Name, Email, Location)

Orders(Order_ID, Purchase_Date)

FK Customer_ID ->Customer(Customer_ID)

ON DELETE NO ACTION, ON UPDATE CASCADE

Product(Product_ID, Name, Description, Price, Quantity_Left)

FK Warehouse_ID -> Warehouses(Warehouse_ID)

ON DELETE SET NULL, ON UPDATE CASCADE

Products_Ordered(<u>Product_ID, Order_ID</u>, Quantity)

FK Order_ID -> Orders(Order_ID)

ON DELETE NO ACTION, ON UPDATE CASCADE

FK Product_ID->Products(Product_ID)

ON DELETE NO ACTION, ON UPDATE CASCADE

Complaints_Table(<u>Unique_Complaint_Number.</u> Date, Passenger_Complaint_Name, Reason_For_Complaint)

FK HR_ID -> HR_Table(HR_ID)

ON DELETE NO ACTION, ON UPDATE CASCADE

FK Costumer_ID -> Customer(Costumer_ID)

ON DELETE NO ACTION, ON UPDATE CASCADE

REPORT

As UPDATE CASCADE means that if the parent primary key is changed, the child value will also change accordingly. Therefore, I think it will be the best of all constrains for UPDATE, because if the foreign key in the child table, that is, primary key of the parent table, will get modified, then at all places where we used that key as foreign key will also get updated automatically.

I have ON DELETE NO ACTION for foreign keys of Complaints_Table, Products_Ordered, Warehouses, Vehicles, Emergency_Contact, Department_Table, and few other other keys. NO ACTION is similar to RESTRICT. That means any attempt to delete the foreign key through parent key table will give an error. This is beneficial as it prevents any data loss that can be caused by any changes in parent table.

In Department_Details, Office_Location, some FKs of Drivers, Warehouse_IDs, and few other Foreign keys I used ON DELETE SET NULL as there is no need of those keys' data if their parent key is deleted in the table. Thus this will set the value of foreign key as NULL in case of deletion of any value in parent table.