AYESHA TABREZ

**Milestone 1 - Database Design**

**COMP 23111**

12/11/2022

**Table of Contents**

1. ERD

* Introduction to ERD
* ERD
* Report outlining design choices

2. Normalisation

* Introduction to ERD
* Relations in 3NF
* Report outlining design choices

3. Relational Schema

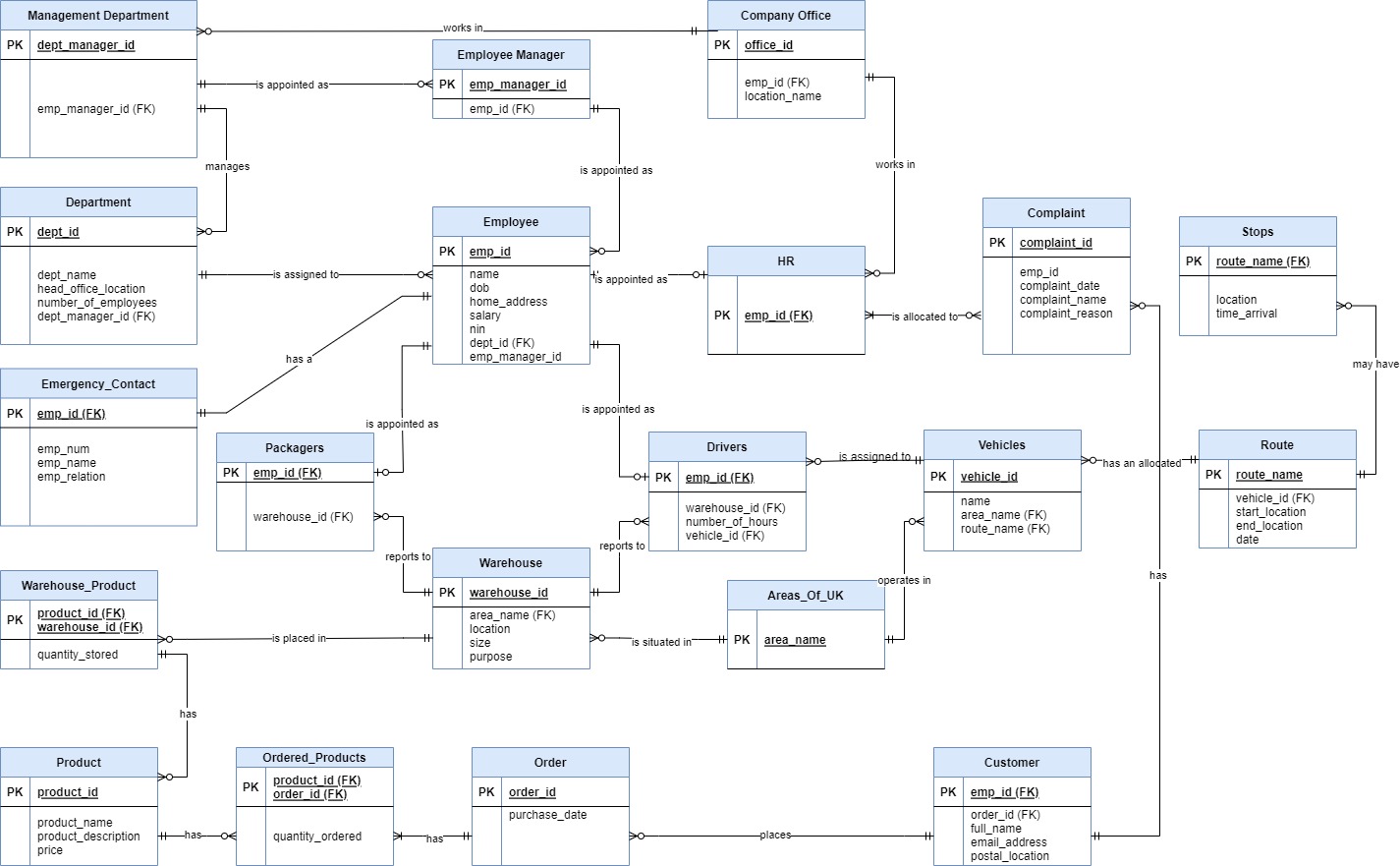
* Introduction to Relational Schema
* Relational Schema
* Report outlining design choices

**INTRODUCTION TO ERD**

ER Diagram stands for Entity Relationship Diagram that displays the relationship of entity sets stored in a database. ER diagrams help to explain the logical structure of databases and are frequently used for conceptual design of database applications. They are created based on 3 basic concepts: entities, attributes and relationships.

I will present my ERD for this coursework below and report the outlining design choices.

**ERD**



**Report Outlining Design Choices**

In my ERD, I have created 20 entities, each having their own attributes.

All entities have simple or atomic attributes which means they cannot be divided into smaller subparts. Each entity has single-valued attributes.

Uniqueness constraint: Each entity has a unique attribute (primary key) which has been underlined and denoted by PK in my ERD. For example, the entity Employee has a primary key emp\_id which uniquely identifies each attribute in the employee entity.

**Composite Attribute**:Note that each Warehouse\_Product can only be uniquely identified by both its product\_id and warehouse\_id. Since each product can be in multiple warehouses, we must know exactly which warehouse product we are referring to and that can only be possible by a combination of product\_id and warehouse\_id that makes it unique. Hence the combination of them is the primary key here.

**Weak Entities:** Emergency\_Contact, Customer, HR, Packagers, Drivers, use foreign key emp\_id in conjunction with their attributes to create a primary key. Warehouse\_Product uses product\_id and warehouse\_id as composite key. Stops uses foreign key route name as its primary key. These entities have *total participation constraint* (existence dependency) with respect to their identifying relationship. So, Emergency\_Contact, Customer, HR, Packagers, Drivers cannot exist without the owner entity: Employee and Stops cannot exist without the owner entity: Routes. For a Warehouse\_Product to exist both warehouse and product must exist.

**ERD Relationships:**In my ERD, the following cardinality constraints are present -:

One to One:Each employee will have exactly one Emergency\_Contact and vice versa.

Drivers, Packagers, HR are uniquely identified by their emp\_id.For instance, every packager will be exactly one employee, but every employee may or may not be a packager. Same for Drivers and HR.

(One to Many) and (Many to One):Every Management\_Department( manager) will work in a specific Office, but every Office can have 0 or many managers working in it. Same for HR.

I have created an entity Employee Manager which is linked to Employee with a one-to-many relationship since every employee will report to a single manager (who is also an employee) but this manager can have multiple employees reporting to him/her multiple departments. This employee manager is appointed as a manager in a management department.Now every department will have a single manager to whom the employee managers will report to.Hence from Management\_Department to Department, we have a one-to-many relationship

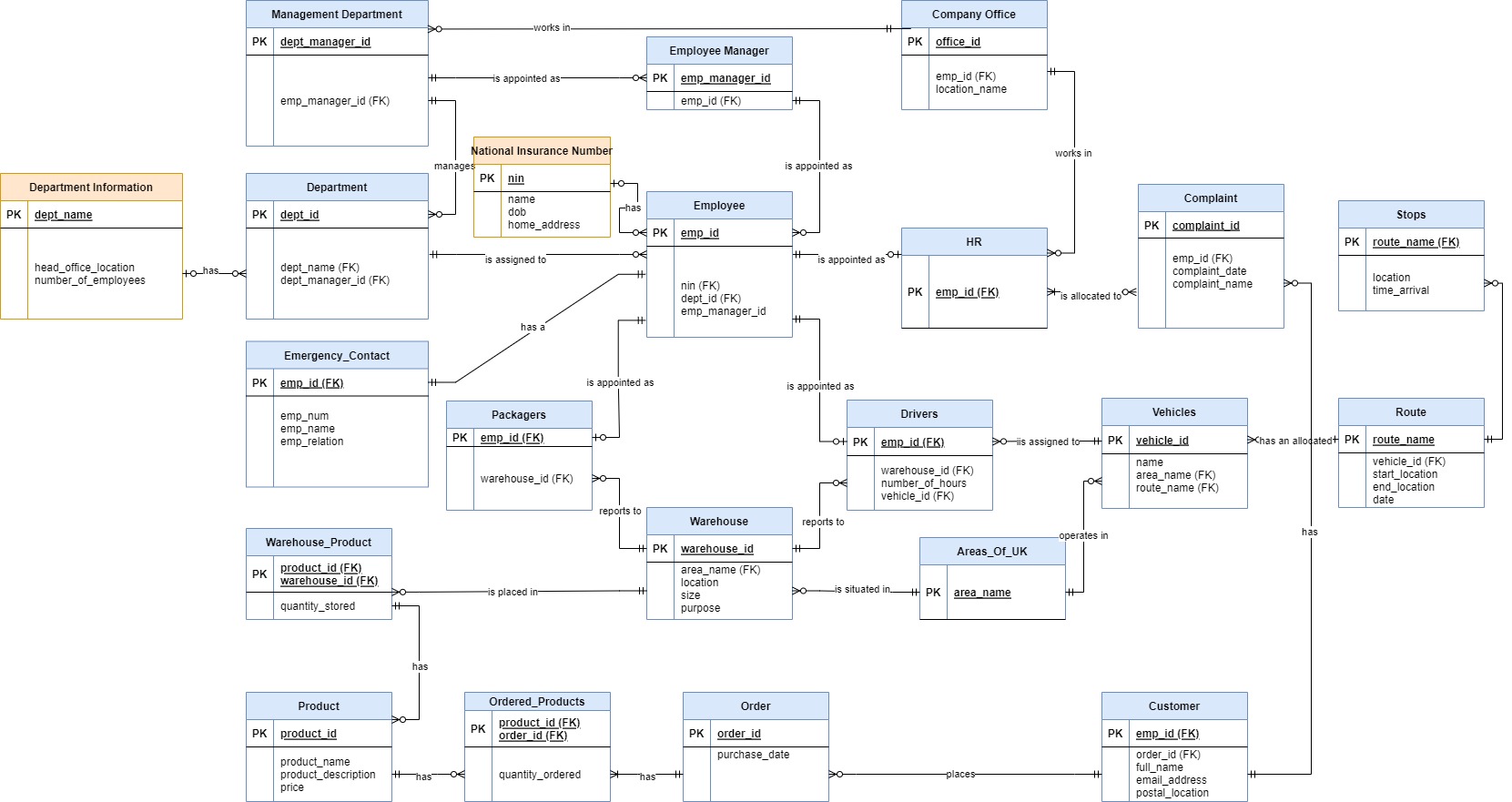
Now I will explain the driver part. Each driver is assigned to a single vehicle, but each vehicle may have 0 or many drivers. Hence between Driver and Vehicle there is (Zero or many) to one (and only one) relationship. Each vehicle has an allocated route on a particular day (so I have date attribute in Route)but each route may have 0 or many vehicles so same relationship between them. Each route may have 0 or more stops but each stop will have exactly one route so the same relationship between stops and route.

Using the same logic, we have implemented the orders and products part with one-to-many relationships between entities created for this part.

**INTRODUCTION TO NORMALISATION**

One of the ways an ERD is enhanced during the logical design phase is through the process of normalization. It is the process of organizing the data in the database, with a focus on minimizing redundancy in relations or set of relations. We will focus on the first 3 (1NF, 2NF, 3NF) for our ERD as they address the 3 data modification anomalies: 1. Insertion Anomaly (Omission of data during insertion due to absence of other data.) 2. Deletion Anomaly (Unintended loss of data due to the deletion of other data.) 3. Update Anomaly (Data inconsistency that results from redundant data and partial updates.)

**NORMALISED ERD**



**RELATIONAL SCHEMA AFTER NORMALISING**

1.Management Department (**dept\_manager\_id**, emp\_manager\_id)

FK emp\_manager\_id → Employee Manager(emp\_manager\_id)

ON DELETE RESTRICT, ON UPDATE CASCADE

2.Employee Manager (**emp\_manager\_id**, emp\_id)

FK emp\_id → Employee(emp\_id)

ON DELETE RESTRICT, ON UPDATE CASCADE

3.Company Office (**office\_id**, emp\_id, location\_name)

FK emp\_id → Employee(emp\_id)

ON DELETE RESTRICT, ON UPDATE CASCADE

4.Areas\_Of\_UK (**area\_name**)

5.Department Information (**dept\_name**, head\_office\_location, number\_of\_employees)

6.Department (**dept\_id**, **dept\_manager\_id**, dept\_name)

FK dept\_manager\_id→ Management Department(department\_manager\_id)

FK dept\_name → Department Information(dept\_name)

ON DELETE RESTRICT, ON UPDATE CASCADE

7.National Insurance Number (**nin,** name, dob, home\_address)

8.Employee (**emp\_id**, nin, dept\_id, emp\_manager\_id)

FK dept\_id → Department(dept\_id)

FK nin → National Insurance Number(nin)

ON DELETE RESTRICT, ON UPDATE CASCADE

9.HR (**emp\_id**)

FK emp\_id → Employee(emp\_id)

ON DELETE CASCADE, ON UPDATE CASCADE

10.Complaint (**complain**t**\_id**, emp\_id, complaint\_date, complaint\_name)

FK emp\_id → Employee(emp\_id)

ON DELETE RESTRICT, ON UPDATE CASCADE

11.Stops (**route\_name**, location, time\_arrival)

FK route\_name → Route(route\_name)

ON DELETE RESTRICT, ON UPDATE CASCADE

12.Emergency Contact (**emp\_id**, emp\_num, emp\_name, emp\_relation)

FK emp\_id → Employee(emp\_id)

ON DELETE CASCADE, ON UPDATE CASCADE

13.Packagers (**emp\_id**, warehouse\_id)

FK emp\_id → Employee(emp\_id)

FK warehouse\_id → Warehouse(warehouse\_id)

ON DELETE CASCADE, ON UPDATE CASCADE

14.Warehouse (**warehouse\_id**, area\_name, location, size, purpose)

FK area\_name → Areas\_Of\_UK(area\_name)

ON DELETE RESTRICT, ON UPDATE CASCADE

15.Drivers(**emp\_id**, warehouse\_id, number\_of\_hours, vehicle\_id)

FK emp\_id → Employee(emp\_id)

FK warehouse\_id → Warehouse(warehouse\_id)

FK vehicle\_id → Vehicles(vehicle\_id)

ON DELETE CASCADE, ON UPDATE CASCADE

16.Vehicles(**vehicle\_id**, name, area\_name, route\_name**)**

FK area\_name → Areas\_Of\_UK(area\_name)

FK route\_name → Route(route\_name)

ON DELETE RESTRICT, ON UPDATE CASCADE

17.Route**(route\_name**, vehicle\_id, start\_location, end\_location, date**)**

FK vehicle\_id → Vehicles(vehicle\_id)

ON DELETE RESTRICT, ON UPDATE CASCADE

18.Warehouse\_Product (**product\_id, warehouse\_id**, quantity\_stored)

FK product\_id → Product(product\_id)

FK warehouse\_id → Warehouse(warehouse\_id)

ON DELETE CASCADE, ON UPDATE CASCADE

19.Product(**product\_id**, product\_name, product\_description, price)

20.Ordered Products**(product\_id, order\_id**, quantity\_ordered)

FK product\_id → Product(product\_id)

FK order\_id → Order(order\_id)

ON DELETE CASCADE, ON UPDATE CASCADE

21.Order (**order\_id**, purchase\_date)

22.Customer (**emp\_id**, order\_id, full\_name, email\_address, postal\_location)

FK emp\_id → Employee(emp\_id)

FK order\_id → Employee(order\_id)

ON DELETE CASCADE, ON UPDATE CASCADE

**Report Outlining Design Choices**

1. 1NF : Our ERD is already in 1NF since all attributes are simple(atomic), there are no multi-valued attributes and all entities have a primary key.

2. 2NF: Our ERD is already in 2NF since:

1. 1NF is satisfied.

2. Every non-key attribute is functionally dependent on the key attribute, hence there is no partial dependency. For example, all non-key attributes in Employee (nin, dept\_id, dept\_manager\_id) depend on the PK emp\_id. Same for all other entities.

3. 3NF: To satisfy 3NF:

1. 2NF must be satisfied. We can see that it is satisfied for our ERD.

2.There are no transitive dependencies for non-key attributes. This condition ***is not*** satisfied and therefore we need to satisfy it to bring it to 3NF form.

**Satisfying 3NF**

In our ERD, for entity Department, I have assumed that all department names are unique. So, we can see there is a functional dependency between dept\_name and head\_office\_location and dept\_name and number\_of\_employees. Thus, there exists a transitive dependency in our ERD which is not good. To remove this, I have created another table Department Information with dept\_name as primary key since all department names are unique and head\_office\_location and number\_of\_employees as the other non-key attributes for this entity. In the Department entity, I have removed head\_office\_location and number\_of\_employees but kept dept\_name which will now be used as a foreign key to access department information.

Our ERD is still not in 3NF. We can see that in the Employee table there is an attribute nin (national insurance number). A nin is unique for every person. Hence, we can see that there is a functional dependency between home\_address, name, dob and nin. Thus, there exists a transitive dependency in our ERD which is not good. To remove this, I have created another table National Insurance Number with nin as the primary key as this will be unique and home\_address,name and dob as the other non-key attributes for this entity. In the Employee table, I have removed home\_address, dob and name but kept nin which will now be used as a foreign key to access National Insurance Number.

Now we can see that there are no more transitive dependencies in any entity. Therefore, we can say that our ERD is now in 3NF normalised form.

**INTRODUCTION TO RELATIONAL SCHEMA**

Relational schema defines the design and structure of the relation like it consists of the relation name, set of attributes/field names/column names. Relational schema can be in tabular form as well as textual form. I will be converting my ERD to relational schema in textual form as shown below:

Relation\_Name (Attribute\_1, Attribute\_2, ... Attribute N)

FK Attribute\_Name → Relation\_Name (Attribute\_Name)

ON DELETE CONSTRAINT, ON UPDATE CONSTRAINT

**RELATIONAL SCHEMA OF NORMALISED ERD**

1.Management Department (**dept\_manager\_id**, emp\_manager\_id)

FK emp\_manager\_id → Employee Manager(emp\_manager\_id)

ON DELETE RESTRICT, ON UPDATE CASCADE

2.Employee Manager (**emp\_manager\_id**, emp\_id)

FK emp\_id → Employee(emp\_id)

ON DELETE RESTRICT, ON UPDATE CASCADE

3.Company Office (**office\_id**, emp\_id, location\_name)

FK emp\_id → Employee(emp\_id)

ON DELETE RESTRICT, ON UPDATE CASCADE

4.Areas\_Of\_UK (**area\_name**)

5.Department Information (**dept\_name**, head\_office\_location, number\_of\_employees)

6.Department (**dept\_id**, **dept\_manager\_id**, dept\_name)

FK dept\_manager\_id→ Management Department(department\_manager\_id)

FK dept\_name → Department Information(dept\_name)

ON DELETE RESTRICT, ON UPDATE CASCADE

7.National Insurance Number (**nin,** name, dob, home\_address)

8.Employee (**emp\_id**, nin, dept\_id, emp\_manager\_id)

FK dept\_id → Department(dept\_id)

FK nin → National Insurance Number(nin)

ON DELETE RESTRICT, ON UPDATE CASCADE

9.HR (**emp\_id**)

FK emp\_id → Employee(emp\_id)

ON DELETE CASCADE, ON UPDATE CASCADE

10.Complaint (**complain**t**\_id**, emp\_id, complaint\_date, complaint\_name)

FK emp\_id → Employee(emp\_id)

ON DELETE RESTRICT, ON UPDATE CASCADE

11.Stops (**route\_name**, location, time\_arrival)

FK route\_name → Route(route\_name)

ON DELETE RESTRICT, ON UPDATE CASCADE

12.Emergency Contact (**emp\_id**, emp\_num, emp\_name, emp\_relation)

FK emp\_id → Employee(emp\_id)

ON DELETE CASCADE, ON UPDATE CASCADE

13.Packagers (**emp\_id**, warehouse\_id)

FK emp\_id → Employee(emp\_id)

FK warehouse\_id → Warehouse(warehouse\_id)

ON DELETE CASCADE, ON UPDATE CASCADE

14.Warehouse (**warehouse\_id**, area\_name, location, size, purpose)

FK area\_name → Areas\_Of\_UK(area\_name)

ON DELETE RESTRICT, ON UPDATE CASCADE

15.Drivers(**emp\_id**, warehouse\_id, number\_of\_hours, vehicle\_id)

FK emp\_id → Employee(emp\_id)

FK warehouse\_id → Warehouse(warehouse\_id)

FK vehicle\_id → Vehicles(vehicle\_id)

ON DELETE CASCADE, ON UPDATE CASCADE

16.Vehicles(**vehicle\_id**, name, area\_name, route\_name**)**

FK area\_name → Areas\_Of\_UK(area\_name)

FK route\_name → Route(route\_name)

ON DELETE RESTRICT, ON UPDATE CASCADE

17.Route**(route\_name**, vehicle\_id, start\_location, end\_location, date**)**

FK vehicle\_id → Vehicles(vehicle\_id)

ON DELETE RESTRICT, ON UPDATE CASCADE

18.Warehouse\_Product (**product\_id, warehouse\_id**, quantity\_stored)

FK product\_id → Product(product\_id)

FK warehouse\_id → Warehouse(warehouse\_id)

ON DELETE CASCADE, ON UPDATE CASCADE

19.Product(**product\_id**, product\_name, product\_description, price)

20.Ordered Products**(product\_id, order\_id**, quantity\_ordered)

FK product\_id → Product(product\_id)

FK order\_id → Order(order\_id)

ON DELETE CASCADE, ON UPDATE CASCADE

21.Order (**order\_id**, purchase\_date)

22.Customer (**emp\_id**, order\_id, full\_name, email\_address, postal\_location)

FK emp\_id → Employee(emp\_id)

FK order\_id → Employee(order\_id)

ON DELETE CASCADE, ON UPDATE CASCADE

**Report Outlining Design Choices**

I have represented my normalised ERD in relational schema textual form above. I have used foreign keys to explain the relationship between relations. The identifiers in ERD are represented as primary keys in our relational schema.

I have used ON DELETE RESTRICT for foreign keys in some relations (wherever I want to be on the safer side). For example, if we try to delete dept\_id from department, the operation will get rejected if one employee at least links on this department.

I have used ON UPDATE CASCADE for all foreign keys in all relations. For example, if we update dept\_id in Department, it will update it accordingly on employees referencing this Department.

For some weak entities, I have used ON DELETE CASCADE since these are weak entities and will not affect any table.

In all many to one relations, I have foreign keys on the one side of the relation.

For example, I will explain the management part.

Every employee reports to a manager. So, in the employee relation I have used emp\_manager\_id as the foreign key which can be used to access Employee Manager relation. In Employee Manager relation, emp\_manager\_id is a primary key but emp\_id is an attribute of this relation. This is because each employee manager can have multiple employees reporting to them, but each employee will report to only one manager. Now each department will have a main department manager. So, in my department relation, I have used dept\_manager\_id as foreign key which can be used to access Management Department. In this dept\_manager\_id is the primary key and this also has emp\_manager\_id as an attribute. This is because there will be only one main department manager but many employee managers reporting to

department manager. I have represented all other relations using foreign keys in a similar manner.

**CITATION(REFERENCES)**

[*https://en.wikipedia.org/wiki/Weak\_entity*](https://en.wikipedia.org/wiki/Weak_entity)

[*https://condor.depaul.edu/gandrus/240IT/accesspages/primary-foreign-keys.htm*](https://condor.depaul.edu/gandrus/240IT/accesspages/primary-foreign-keys.htm)

[*https://tdan.com/normalizing-with-entity-relationship-diagramming/4583#:~:text=Normalization%20utilizes%20association%20among%20attributes,the%20conceptual%20data%20modeling%20phase*](https://tdan.com/normalizing-with-entity-relationship-diagramming/4583#:~:text=Normalization%20utilizes%20association%20among%20attributes,the%20conceptual%20data%20modeling%20phase)*.*

[*https://medium.com/@ashleylynnrapone/database-design-from-entity-relationship-diagrams-to-physical-diagrams-16da210d01e9*](https://medium.com/@ashleylynnrapone/database-design-from-entity-relationship-diagrams-to-physical-diagrams-16da210d01e9)

*UNIVERSITY OF MANCHESTER: DATABASE SYSTEMS SLIDES.*