LAU
School of Arts and Sciences

DATABASE DESIGN FOR A NEWSPAPER

PHASE II

By

SQL SQUAD

HANINE ZOUHEIR AL KHATIB

JAMIL MOHAMMAD AWADA

MOHAMMAD HASSAN DHAINI

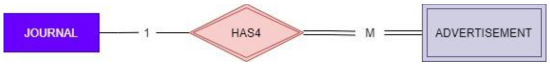
MOHAMMAD YATEEM

Report submitted to: **Dr. Ramzi Haraty**

I-Table of Contents

Changes to Phase 1:

- ER Diagram:
  - Added “number of hours” attribute to “WORKS_FOR” relationship between EMPLOYEE entity and DEPARTMENT
  - Relationship of Journal and Advertisement reversed total participation on both sides, from partial to total on advertisement side and total to partial from journal side.



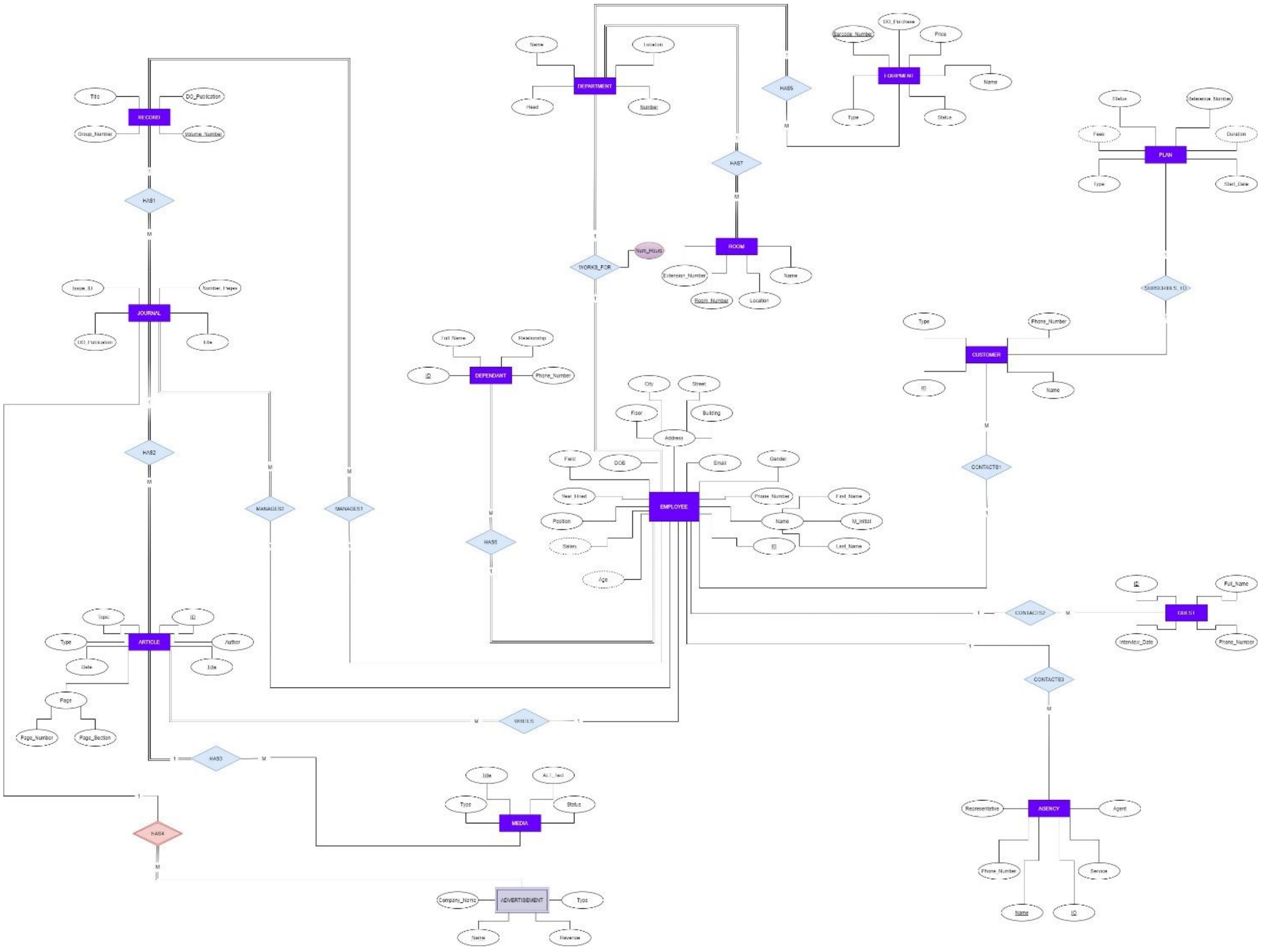
  - Room changed from weak to strong entity type
  - “Has” relationship now numbered
  - “Manages” relationship now numbered
  - “Contacts” relationship now numbered
- Report:
  - Elaborated on the introduction section
  - Fixed ROOM entity type description
  - Added relational schemas to the report.
  - Improved table of content’s structure and content.

Introduction

Our team, named “The SQL Squad” (a name inspired by “The Suicide Squad”, a DC Universe team of villains), are delighted to submit our report detailing the thought process and design choice justifications for our hypothetical newspaper organization. Our members have visited “Al Nahhar” Newspaper HQ located in Downtown, Beirut in order to inquire about their Database outline and implementation techniques, as well as their organizational management. There, we met with Elie Bou Moussa, the IT manager responsible for the upkeep and maintenance of the organization’s IT services, as well being the individual responsible for communicating with their external database service provider. Elie was a great help in giving us a tour of the campus grounds, explaining the organizational chart and employee hierarchy along the way. We were shown multiple departments and had the pleasure to ask department managers and employees about their place in the company as well as the responsibilities they must fulfill. It is without a doubt that we accredit much of our success in accurately representing a newspaper to the generous and hospitable team at Al Nahhar Newspaper. On our efforts, the experience working as a team as well as going through a creative process has proven to be an undeniably helpful asset in each of our members' repertoire. Each team member has grown better as a scientist as well as a free thinker after having went through a methodical research process and applying their knowledge to create and deploy a system as complex and rich as ours.

This report will go over our work process, with a thorough dive into our reasoning behind every design choice as well as their function in the overall final project. We had an idea in mind in regard to the approach we will be taking throughout this project. We followed the plan and stuck to a methodical way of handling the project, going through each concept one by one and expanding on it. Much like a tree, this project sprouted from a collection of simple notions and ideas that individually grew to form the collective, each with its own time and care put into it. Regarding software, our team has opted to use "Draw.io", an ER-Diagram modeling software freely provided by Diagrams.net. We have decided to use this software for it's ease of use, accessibility, widespread compatibility and 0$ price tag. Other alternatives such as Microsoft Visio were considered but were ultimately overshadowed due to complications in installation and unwanted bloat. Nevertheless, we believe that the project wasn't undermined by any of these factors and that we were able to deliver the best product possible with the tools at hand and the experiences under our belt. Lastly, we thank Dr. Ramzi Harati for guiding us and teaching us everything we need to know to accomplish such a job. We hope the project is up to par with expectations and meets the requirements of a fine SQL-Based Database.

ER Diagram for the newspaper



System Description

In the digital age, man seldom feels the warmth of freshly printed newspapers chock full of today's latest news and events. Previously one of the few and most common ways to acquire information, newspapers have since been overshadowed by their digital counterparts, which offer a faster, cheaper, more convenient way of catching up on the daily. Nevertheless, newspapers remain journalists' and political and economic figures main way of influence and are a staple in many people's morning routine.

Our employees are split into multiple specializations that interact with each other to deliver a quality final product. The organization is split into departments, with each employee working in a specific department equipped with its own tools and resources specific to its own function. Each department has a name, location, number and head. Each department has specific rooms as well as its own equipment specific to its function. Employees contact customers, agencies, and guests. Most importantly, employees produce journals, which are the core of the newspaper and an integral part in our project.

Journals are the newspaper itself, a collection of pages and information neatly organized and decorated. Journals consist of pages, of which can be articles, different types of media, and advertisements. A record of a journal is kept and stored in an archive along with other information.

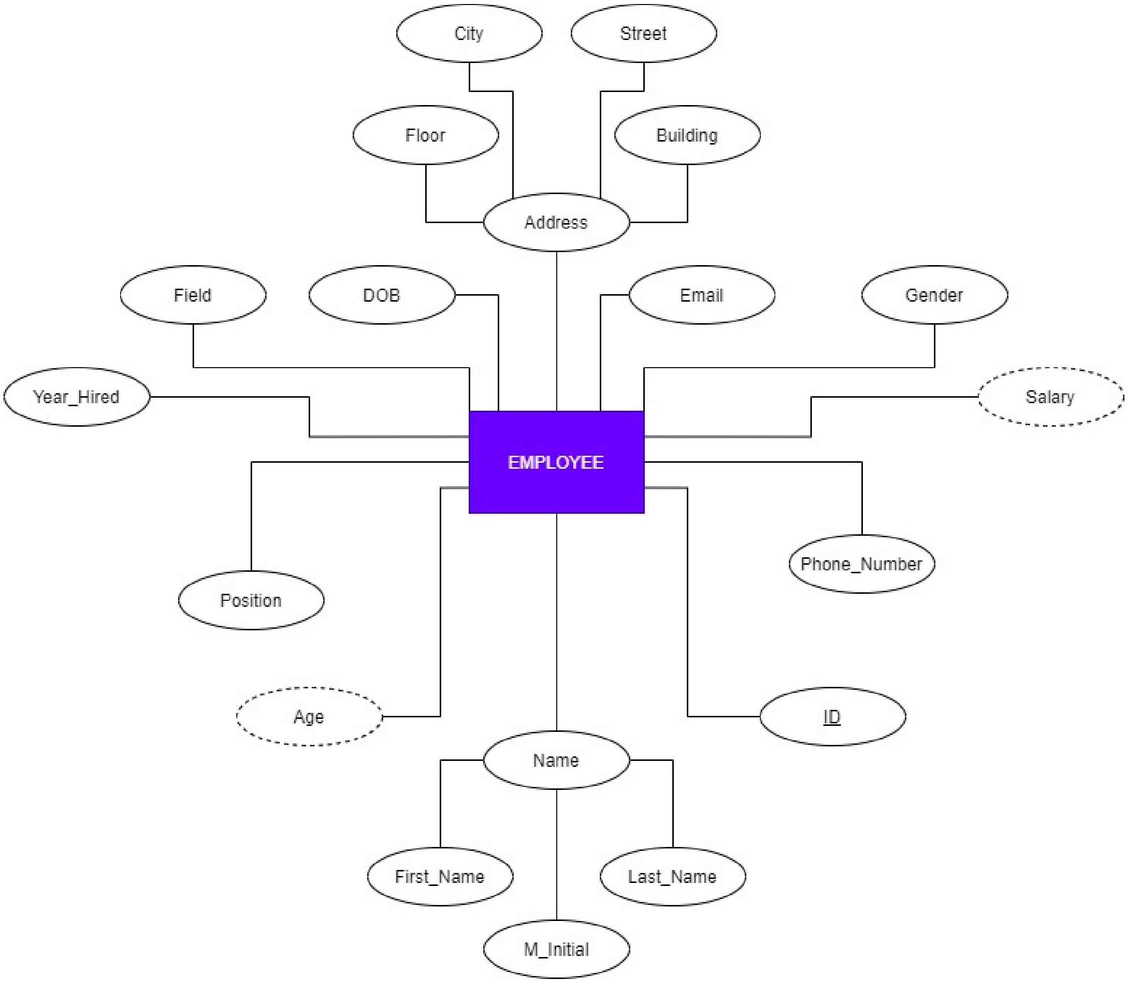
Employee types and responsibilities include photography, editing, writing, illustrating, and other miscellaneous jobs. Employees have equipment they use that either belongs to them or the studio, with each piece of equipment being exclusive to a certain job. Employees have dependents they rely on for emergency contact, or if the employee couldn’t be reached.

Of course, no newspaper is complete without its foreign agents: Customers, agencies, and guests. They are the outside forces that act on the newspaper and influence its profits and influence. Customers form the main source of revenue for the newspaper, and they could be of any type or quantity. Agencies are a way for the newspaper to get media and news, as well as any other service it might need. Guests are often invited to provide insight on topics, to be held interviews with, and to have their work showcased.

Without further ado, we, “the SQL squad”, present to you our diagram, containing our entity types and relationships, for our newspaper.

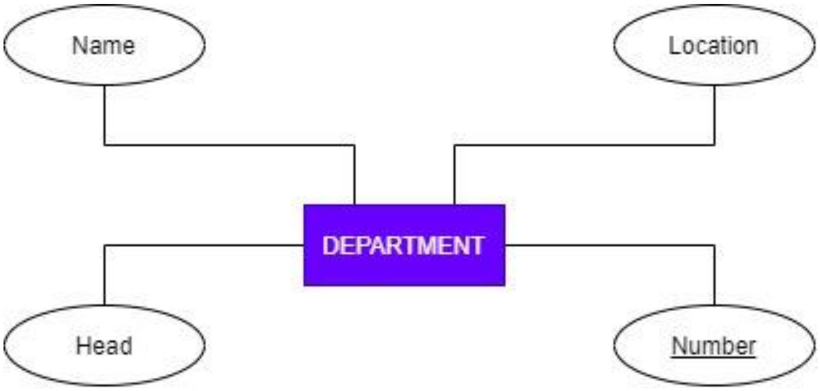
Entity Types:

1- Employee



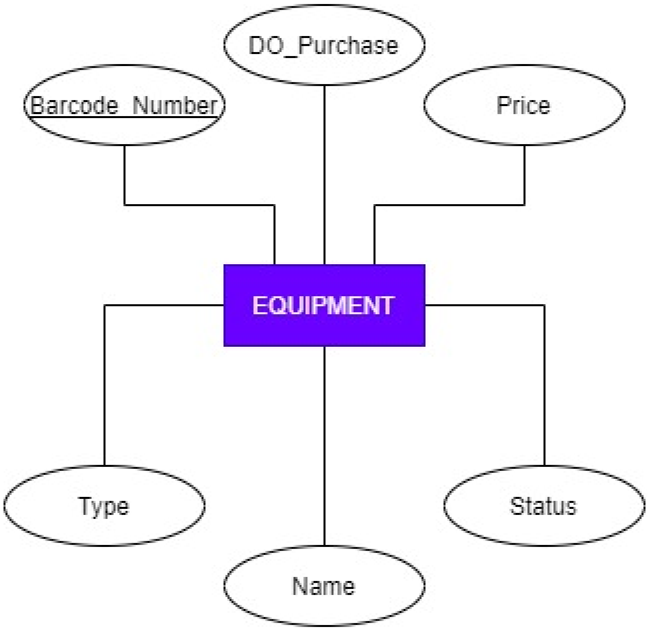
Employees are the most important customers given that they can provide crucial insights into the overall customer experience, although they are often overlooked or neglected. An employee is identified by a unique **ID** which is considered the key in this entity. Every doctor has a name (composite attribute) which is divided into last name, first name, and middle initial. Each employee has also other attributes that describe them, including gender, date of birth, age which is derived from the latter, phone number including the country code then the n-digit number. Each employee has other attributes like email, year hired, and field which indicates what he works as, position indicating the level he has reached in his field, and composite address of the employee's residence. There are multiple departments that an employee can work for, that is why a field is provided. The phone number of the employee is provided to ensure contact in case of any ordeal faced. The salary is a derived attribute that is provided from both the field and level reached of each employee and the number of worked hours in the department.

2- Department



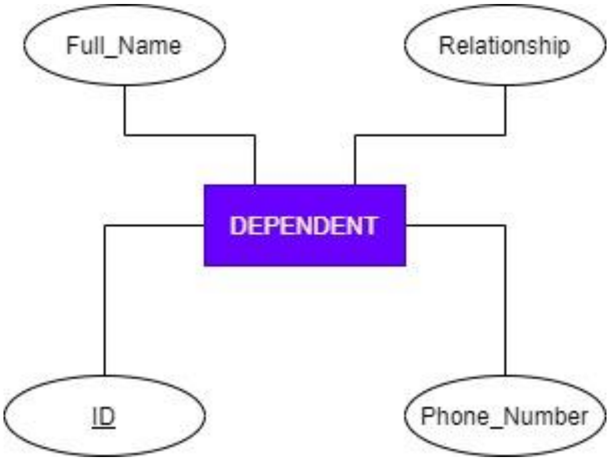
A newspaper has several departments that are always active and efficient, much like a beehive. In a newspaper, we may have many departments that specialize in a specific domain like Finance, Human Resources, Information Technology, Administrative, Printing, Editorial or Advertising. Each department is represented by its head and has as key a **number**, a name, a phone number as well as an email to contact each department, and a location for easier navigation.

3- Equipment



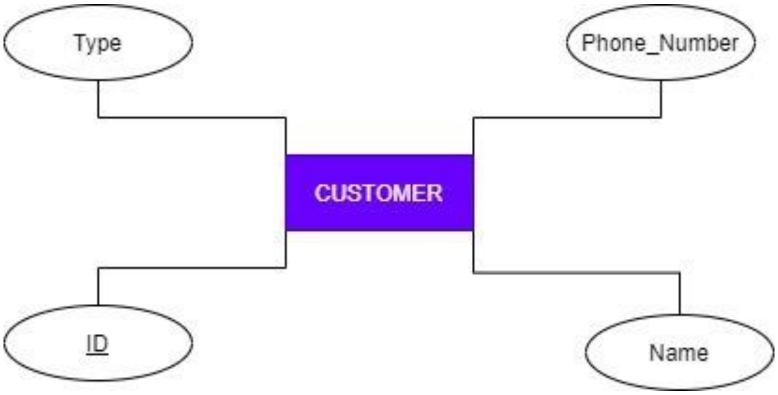
Every newspaper company needs equipment like printers, cameras, and computers to facilitate the work of the reporters, photographers, journalists and the rest of employees. Every piece of equipment has its **barcode number** for references of the product (key), a type (Camera, Microphone, Light, Wallpaper, etc…) and has a date of purchase to know when the product was bought as well as a price that informs us how much each piece of material cost. Every piece of equipment has a status indicator indicating whether it's in use, free, stockpiled, in repair, sold, in transit, etc…

4-Dependent



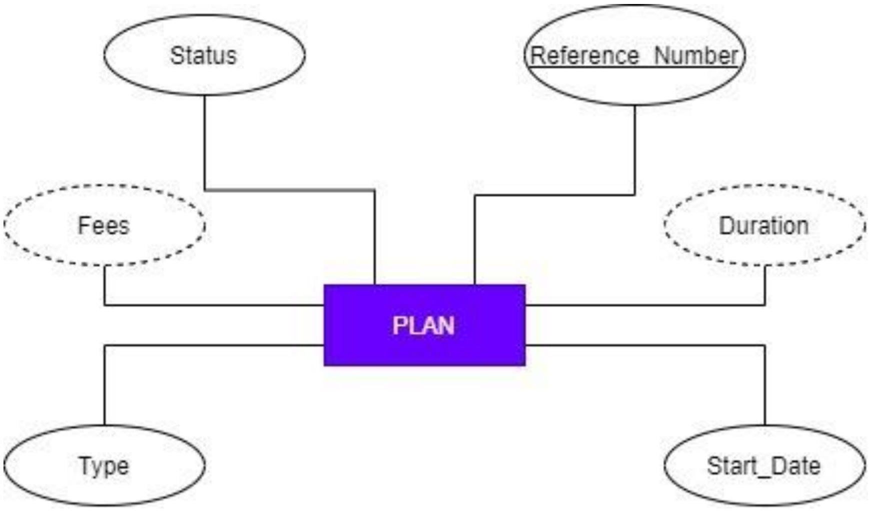
Dependents are the emergency contacts employees depend on in case of an emergency or the organization loses contact with staff. They have very simple attributes such as the key **ID**, a name, phone number, email and their relationship with the employee. It is important to have someone as a backup in case something goes awry.

5-Customer



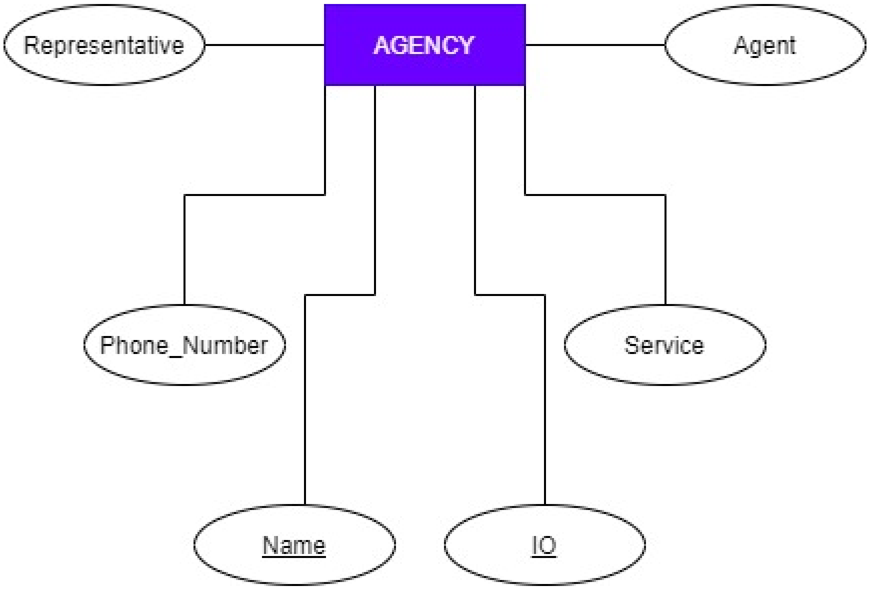
A customer is any individual or organization that wishes to subscribe to the newspaper and receive weekly, biweekly or monthly journals. Every customer is contacted and contacts an employee in order to manage their subscription (or set up one.). Customers have a name, **ID** as a key for storage, phone number for contact, and a type. Customers can be individual people, retailers, or any other entity. They subscribe to a plan that dictates how often and how they receive newspapers. Customers form the vast majority of income for the organization and are a key component to keep track of in order to maximize profit and engagement.

6- Plan



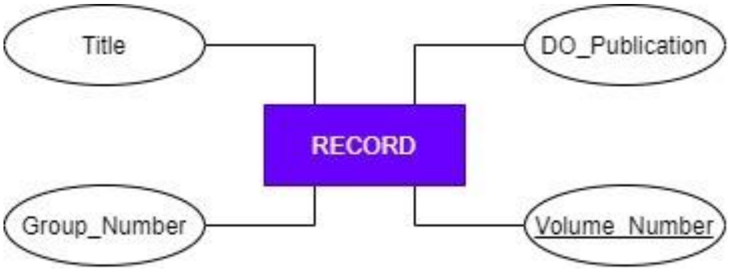
Plans are the structure in which the product gets delivered and sold. They are the basis for which our customers receive their promised newspapers. Plans have a type, with differing perks and pricing for each type, as well as fees such as subscription and delivery fees (derived from the type of plan subscribed to). Plans have a start date and continue up until they are terminated with the derived attribute, duration. They are referenced by a **reference numbe**r as a key and have a status that indicates whether the plan is active, inactive, or on hold.

7- Agency



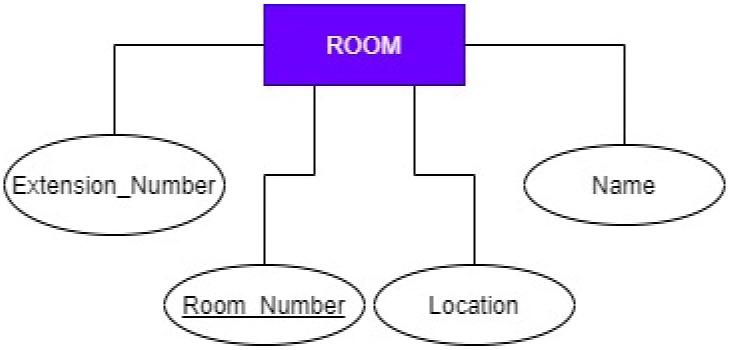
Agencies are the other major organization that the newspaper will be making contact with. They are most often a larger source of media and content used in the papers themselves. Agencies have a representative, i.e., a person that the newspaper will be regularly contacting for sales, meetings, and such. Not to be confused with representatives, agents are agency staff that deliver the product and services: They are managed by the representative and instructed to provide services and products to the newspaper in accordance with the contract. Agencies have an **insertion order** as primary key as well as a *name* as secondary key and phone number for contact. Every agency provides a specific service, such as photo delivery, information gathering, investigative work, etc.…

8-Record



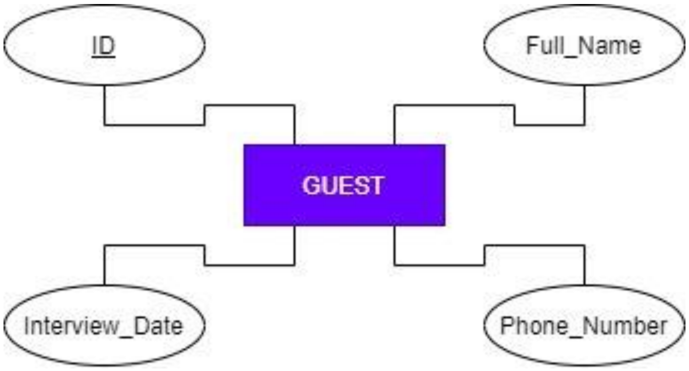
It is crucial for journals to have records kept in case a recall, revision, or review is needed. Records act as symbolic monuments for the newspaper as well, indicating its history and past successes and failures. “History repeats itself” as they say. Records have a title, a date of publication, and are stored in groups. These groups can be categorized into time, topic, political events, or other criteria. Records have a **volume number** as key to identify them within these groups.

9-Room



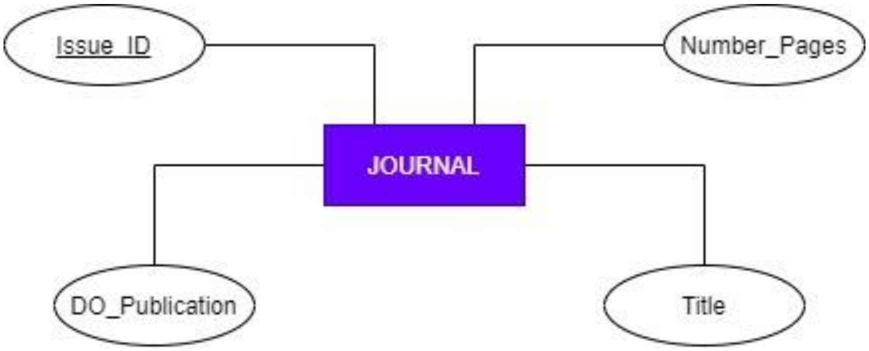
A room can be simply defined as the area in which productive activity takes place or where other miscellaneous affairs are held. A room has a name, an extension number for people to call, a **room number** for quick identification to distinguish rooms, and a location for easier navigation to know in which building or area it is located. It is very important for an organization to have organized rooms, to be able to maintain a study workflow free of disruptions.

10- Guest



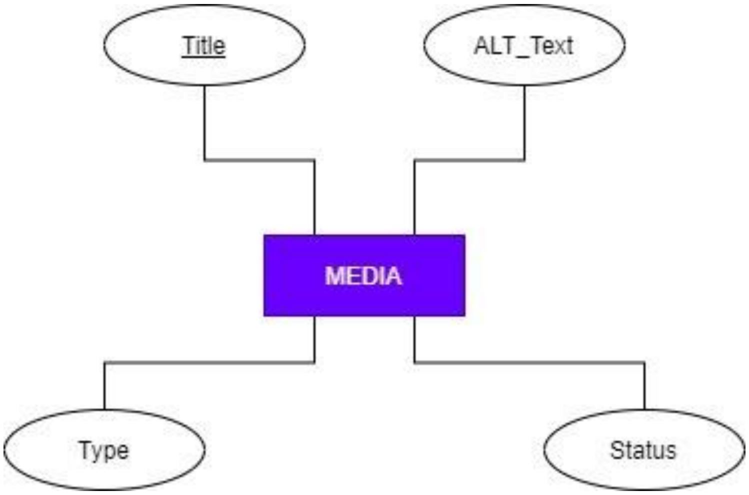
A guest is a person who would come to the newspaper for specific reasons that are of value for the newspaper or personal reasons. That’s why each guest has a Motif, i.e., a reason for coming, an **ID** as key to keep track of every guest, a name, and a phone number. Guests constantly show up for news coverage, interviews, a speech or a presentation. Guests are kept track of incase they are recurring, or if media with guests needs to be found.

11- Journal



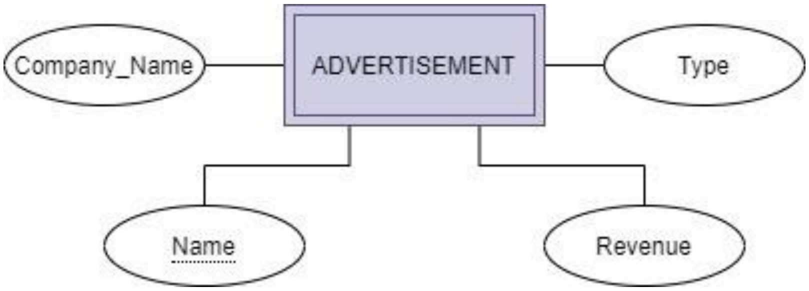
A journal is any one issue of a newspaper. It is every instance of the collective media and news produced on any given day. A journal has a date of publication, a key which is the **issue ID**, a title, and number of pages. Journals are the core product of a newspaper, with all productive efforts being concentrated onto their production and distribution, as well as upholding a minimum standard of quality. In fact, journals are so important that records are kept of every journal.

12- Media



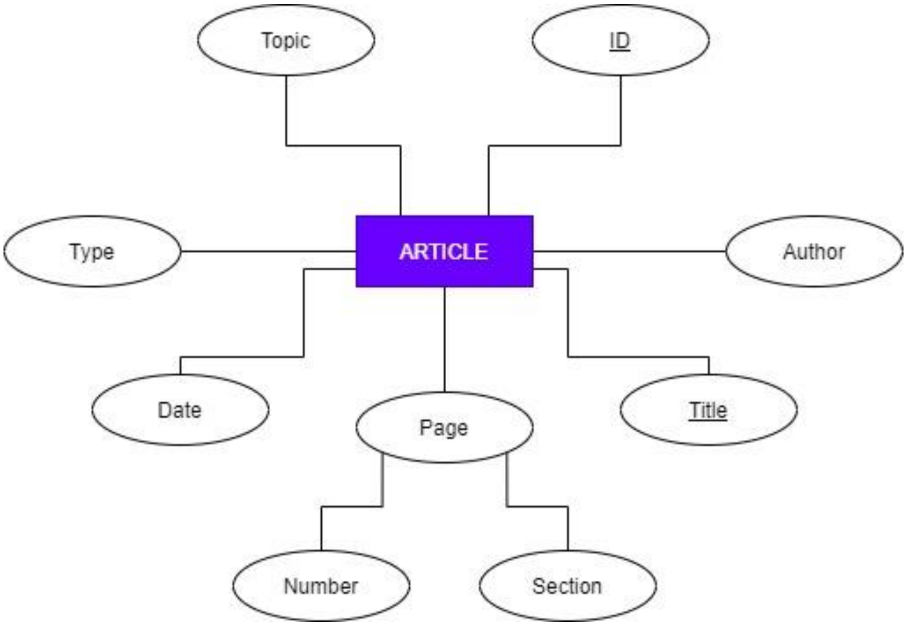
Media refers to the collective imagery and audio used in day-to-day issues to represent abstract ideas or concrete objects. Seeing something is different from reading about it, so it is highly important a large supply of media is constantly being refreshed and kept up to date. Every piece of media has a type (Photo, Video, Drawing, Audio, etc.…), as well as a key, a **title**. The status refers to the state the piece of media is in, as it could be in use, inactive or archived. Alt-text refers to a small piece of text to quickly give whoever wishes to access it a quick brief description.

13- Advertisement



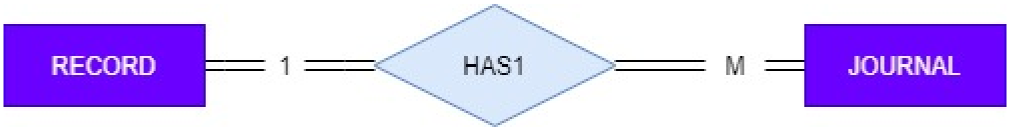
Advertisements are another source of income for the newspaper. They are placed by advertisers and could be of multiple types. Advertisements have a *name* as a weak key (rendering the advertisement entity as weak) to describe the advertisement itself (e.g., 2004 Ford for sale) as well as the seller's name and the amount of revenue retrieved from the advertisement itself. Advertisements constitute a small to medium portion of the journal.

14- Article

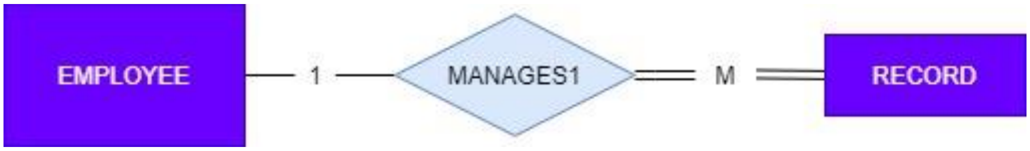


Articles are the core of newspapers considering that these articles can provide a useful source of information about historical and current events. Each article has an **ID** (key), a specific topic, an author who is an employee in this newspaper company, date of writing this article, a title which is a secondary key, and pages which is a composite attribute consisting of the page number and section of group of pages.

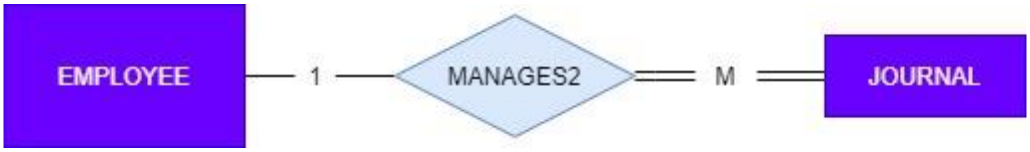
Relationships:



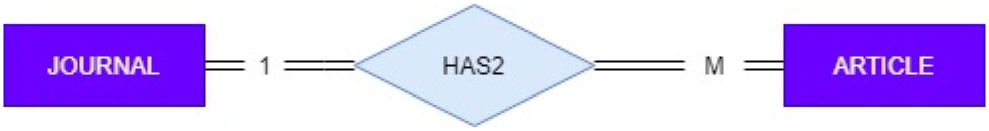
Each **RECORD** has journals saved in the database of newspaper company annually. Thus, a "**HAS**" relationship is created between **RECORD Entity** and **JOURNAL Entity**. The relationship is 1/Many since each record can have multiple journals. The participation is total on both sides since every record has multiple journals and every journal exists in some record. Every record has multiple journals, yet each journal is that of a specific single record.



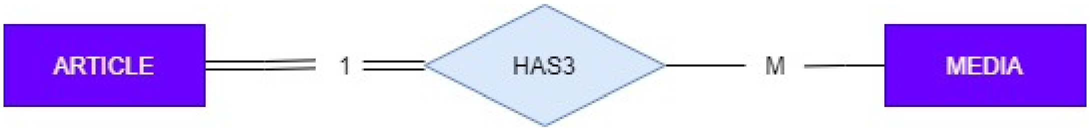
Each **EMPLOYEE** manages records saved in the database of newspaper company. Thus, a "**MANAGES**" relationship is created between **EMPLOYEE Entity** and **RECORD Entity**. The relationship is 1/Many since each employee can manage multiple records. The participation is total on record side since every record is managed by some employee, yet it is partial on employee side since not all employees take care of managing records.



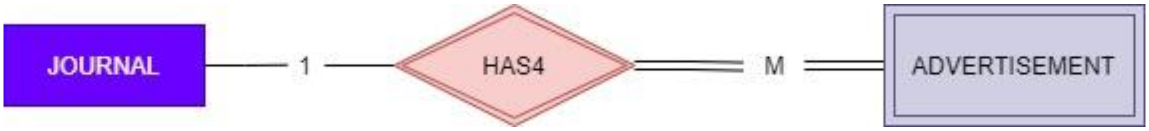
Each **EMPLOYEE** manages journals saved in the database of newspaper company. Thus, a "**MANAGES**" relationship is created between **EMPLOYEE Entity** and **JOURNAL Entity**. The relationship is 1/Many since each employee can manage multiple journals. The participation is total on journal side since every journal is managed by some employee, yet it is partial on employee side since not all employees take care of managing journals.



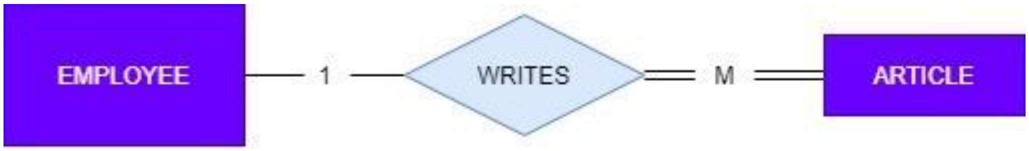
Each **JOURNAL** has articles saved in the database of newspaper company. Thus, a "**HAS**" relationship is created between **JOURNAL Entity** and **ARTICLE Entity**. The relationship is 1/Many since each journal can have multiple articles. The participation is total on both sides since every journal has multiple articles and every article exists in some journal. Every journal has multiple articles, yet each article is that of a specific single journal.



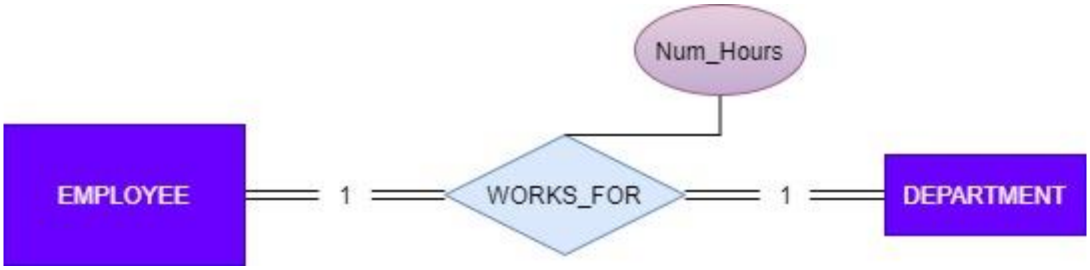
Each **ARTICLE** has media saved in the database of newspaper company. Thus, a "**HAS**" relationship is created between **ARTICLE Entity** and **MEDIA Entity**. The relationship is 1/Many since each article can have multiple medias. The participation is total on article side since every article has multiple different medias, yet it is partial on media side since not all medias exist in an article. Every article has multiple medias, yet each media is that of a specific single article.



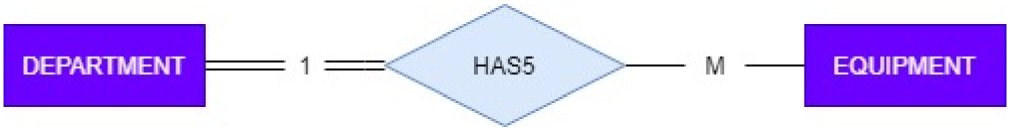
Each **JOURNAL** has advertisements saved in the database of newspaper company. Thus, a "**HAS**" identifying relationship is created between **JOURNAL Entity** and **ADVERTISEMENT Weak Entity**. The relationship is 1/Many since each journal can have multiple advertisements. The participation is total on journal side since every journal has multiple advertisements, yet it is partial on advertisement side since each advertisement is that of a specific single journal. Every journal has multiple advertisements, yet not all advertisements exist in a journal.



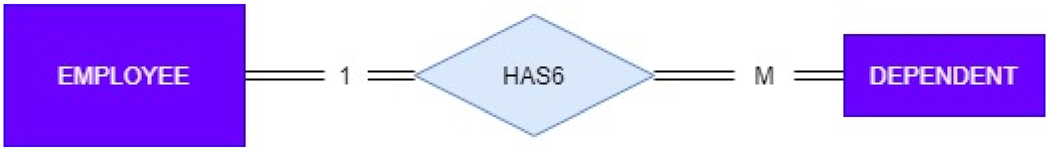
Each **EMPLOYEE** writes articles saved in the database of newspaper company. Thus, a "**WRITES**" relationship is created between **EMPLOYEE Entity** and **ARTICLE Entity**. The relationship is 1/Many since each employee can write multiple articles. The participation is total on article side since every article is written by some employee, yet it is partial on employee side since not all employees take care of writing articles.



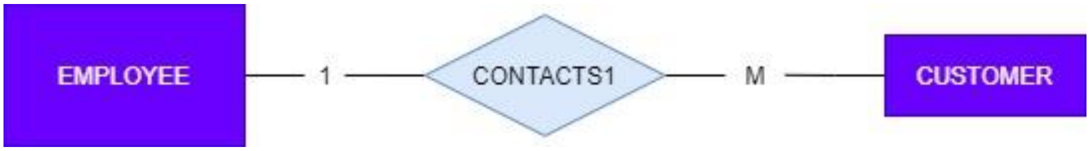
Each **EMPLOYEE** works in a certain field in the newspaper company under a certain department. Thus, a "**WORKS_FOR**" relationship is created between **EMPLOYEE Entity** and **DEPARTMENT Entity**. The relationship is 1/1 since each employee can work for a single department only. The participation is total on both sides since every employee works in a department and every department is worked in. Every employee works in exactly a single department and every single department has specific employees that work in it. The relationship has an attribute: the number of hours which states how many hours the employee works in each department to calculate their salary (as a derived attribute to EMPLOYEE entity).



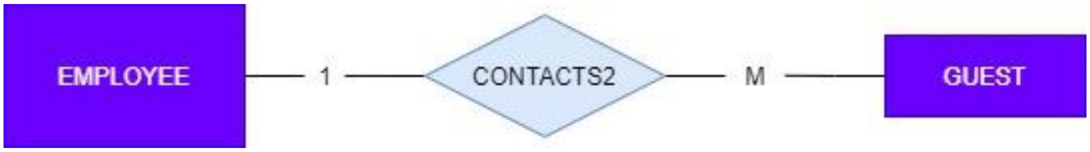
Each **DEPARTMENT** has equipment needed by the employees to perform their tasks. Thus, a "**HAS**" relationship is created between **DEPARTMENT Entity** and **EQUIPMENT Entity**. The relationship is 1/Many since each department can have much equipment. The participation is total on department side since every department has equipment yet is partial from equipment side since not all equipment is found in a certain department.



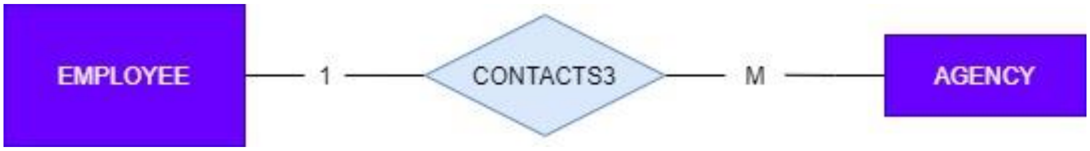
Each **EMPLOYEE** has dependents saved in the database of newspaper company so that if by any chance an accident occurs the department employees would be able to contact their emergency contacts. Thus, a "**HAS**" relationship is created between **EMPLOYEE Entity** and **DEPENDENT Entity**. The relationship is 1/Many since each employee can have many dependents. The participation is total on both sides since every employee has a dependent and every dependent in the database is that of an employee. Every employee has multiple dependents, and each dependent is that of a specific single employee.



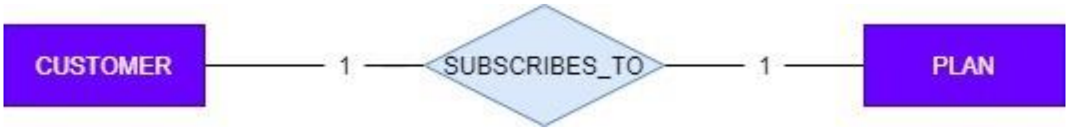
Each **EMPLOYEE** contacts customers to review subscription to plans. Thus, a "**CONTACTS**" relationship is created between **EMPLOYEE Entity** and **CUSTOMER Entity**. The relationship is 1/Many since each employee can contact many customers. The participation is partial on both sides since not all employees are required to contact customers, and not all customers are contacted by employees. Every employee contact multiple customers, and each customer is contacted by a single employee.



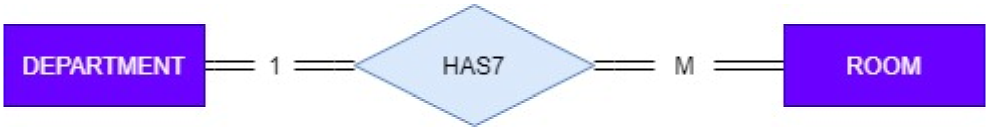
Each **EMPLOYEE** contacts guests for interviews, collaboration, or meetings. Thus, a "**CONTACTS**" relationship is created between **EMPLOYEE Entity** and **GUEST Entity**. The relationship is 1/Many since each employee can contact many guests. The participation is partial on both sides since not all employees are required to contact guests, and not all guests are contacted by employees. Every employee contacts multiple guests, and each guest is contacted by a single employee.



Each **EMPLOYEE** contacts agencies in case the newspaper company needs any external resources. Thus, a "**CONTACTS**" relationship is created between **EMPLOYEE Entity** and **AGENCY Entity**. The relationship is 1/Many since each employee can contact many agencies. The participation is partial on both sides since not all employees are required to contact agencies, and not all agencies are contacted by employees. Every employee contacts multiple agencies, and each agency is contacted by a single employee.



Each **CUSTOMER** can subscribe to specific plans offered by the newspaper company. Thus, a "**SUBSCRIBES_TO**" relationship is created between **CUSTOMER Entity** and **PLAN Entity**. The relationship is 1/1 since each customer can subscribe to one plan only. The participation is partial on both sides since not all customers subscribe to plans, and not all plans can be subscribed to at the same time. Every customer subscribes to a single plan, and each single plan is subscribed by a single customer.



Each **DEPARTMENT** has rooms in some location of newspaper company. Thus, a “**HAS**” identifying relationship is created between **DEPARTMENT Entity** and **ROOM Weak Entity**. The relationship is 1/Many since each department can have multiple rooms. The participation is total on both sides since every department has multiple rooms, and all rooms exist in a certain department.

ER to Relational Mapping Algorithms:

After designing the ER schema and having displayed the database for Promise Hospital as a system of entities, attributes, and relationships, this high-level design must be translated into a relational database design. To map the ER design to a relational database design, a seven-step algorithm needs to be followed. The following is a detailed description on applying the different steps to our database design. The steps in brief are as follows:

- **Step 1:** All regular entity types are mapped into relations schemas. By regular, we mean that only non-weak entities will be mapped in this step. For every regular entity, only the simple attributes are encoded into the relation schemas. Composite attributes are broken down into their simple attribute components. Multivalued and derived attributes are not encoded in this step. Multivalued attributes will be added in the Step 6.
- **Step 2:** All weak entity types are mapped into relation schemas. As in Step 1, only the simple attributes are encoded into the relation schemas. Composite attributes are broken down into their simple attribute components. Multivalued and derived attributes are not encoded in this step. Multivalued attributes will be added in the Step 6.
- **Step 3:** All binary 1:1 relationship types are mapped into relation schemas. Specifically, in this step, we apply the foreign key approach where we choose the entity on the total participation side of the relation, then we add as a foreign key the primary key of the other entity participating in this relation.
- **Step 4:** All binary 1:N relationship types are mapped into relation schemas. As in Step 3, we apply the foreign key approach. We add a foreign key in the entity type at the many sides of the relationship. This foreign key will be the primary key of the other entity type participating in this relationship.
- **Step 5:** All binary M:N relationship types are mapped into relation schemas. Unlike in Step 3 and 4, we encode the relationships by creating a new relation which includes, as foreign keys, the primary keys of all participating relations. Their combination would form the primary key of this newly created relation. This step is not represented here in our report because the database does not include any M:N relationship types.
- **Step 6:** All multivalued attributes that were left over from the previous steps are mapped. Specifically, a relation is created for every multivalued attribute. It would contain the

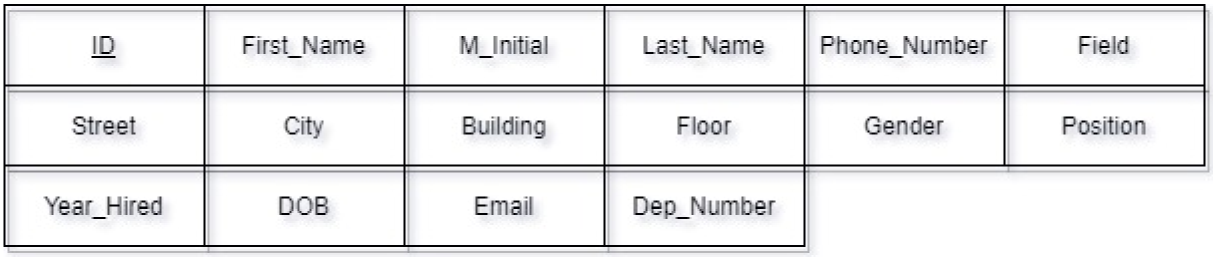
primary key of the entity has that attribute. This step is not represented here in our report because the database does not include any multivalued attributes.

- **Step 7:** All N-ary relationship types are mapped in this step. But there are no N-ary relationships in the database, so nothing is done for this step in this database system.

STEP 1: Mapping of Regular Entity Types

In the first step, the regular entity types must be mapped into relations. Each regular entity is going to have its own relation that includes all of its simple attributes and a single primary key which is underlined. The regular (strong) entities in this database design for Promise Hospital are EMPLOYEE, DEPARTMENT, DEPENDENT, AGENCY, ARTICLE, ADVERTISEMENT, CUSTOMER, GUEST, PLAN, RECORD, JOURNAL, MEDIA, EQUIPMENT, ROOM.

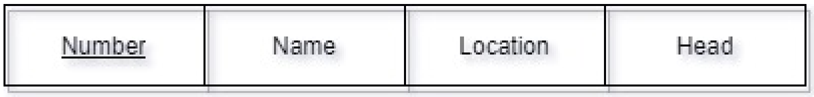
1- EMPLOYEE:



ID	First_Name	M_Initial	Last_Name	Phone_Number	Field
Street	City	Building	Floor	Gender	Position
Year_Hired	DOB	Email	Dep_Number		

The EMPLOYEE entity contains simple, derived, composite and multivalued attributes. The derived attribute *Age* and *Salary* are not represented in this relation. This relation only includes all simple attributes and the primary key **ID** which is underlined. The EMPLOYEE entity has *Name* as a composite attribute of which only the simple attributes *First_Name*, *Last_Name*, and *M_Initial* are included in the relation. This entity also has *Address* as a composite attribute of which only the simple attributes *Street*, *City*, *Building*, and *Floor* are included.

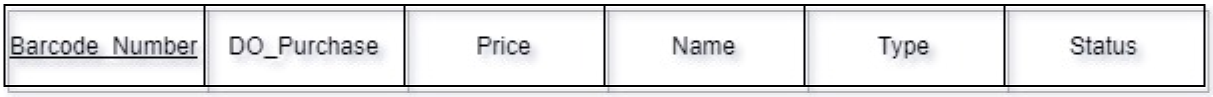
2- DEPARTMENT:



Number	Name	Location	Head

The DEPARTMENT entity contains simple attributes only, so we simply include in this relation the four attributes: Name, Location, Head, and **Number** which is underlined because it is a primary key.

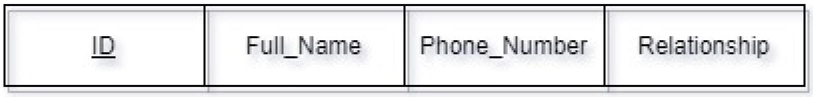
3- EQUIPMENT



Barcode_Number	DO_Purchase	Price	Name	Type	Status

The EQUIPMENT entity contains simple attributes only, so we simply include in this relation the six attributes: Name, Type, Price, Status, DO_Purchase, and **Barcode_Number** which is underlined because it is a primary key.

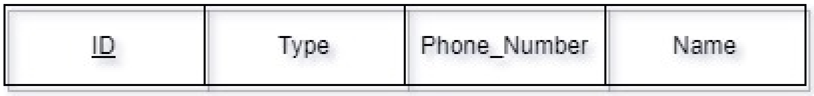
4- DEPENDENT:



ID	Full_Name	Phone_Number	Relationship

The DEPENDENT entity contains simple attributes only, so we simply include in this relation the four attributes: Full_Name, Relationship, Phone_Number, and **ID** which is underlined because it is a primary key.

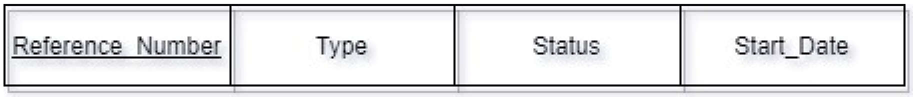
5- CUSTOMER:



ID	Type	Phone_Number	Name

The CUSTOMER entity contains simple attributes only, so we simply include in this relation the four attributes: Name, Type, Phone_Number, and **ID** which is underlined because it is a primary key.

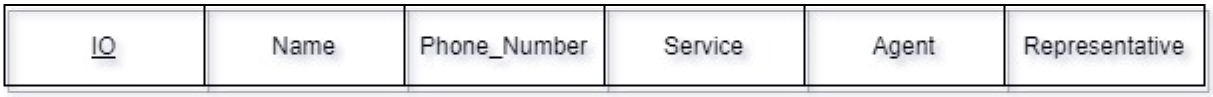
6- PLAN:



Reference_Number	Type	Status	Start_Date

The PLAN entity contains simple attributes only, so we simply include in this relation the four attributes: Type, Status, Start_Date, and **Reference_Number** which is underlined because it is a primary key. The PLAN entity also has the derived attributes *Fees* and *Duration* but are not represented in this relation.

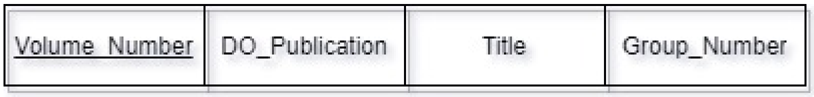
7- AGENCY:



IO	Name	Phone_Number	Service	Agent	Representative

The AGENCY entity contains simple attributes only, so we simply include in this relation the six attributes: Name, Service, Agent, Representative, Phone_Number, and **IO** which is underlined because it is a primary key.

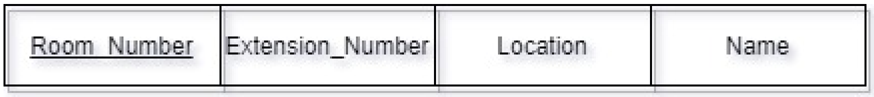
8- RECORD:



Volume_Number	DO_Publication	Title	Group_Number

The RECORD entity contains simple attributes only, so we simply include in this relation the four attributes: Title, DO_Publication, Group_Number, and **Volume_Number** which is underlined because it is a primary key.

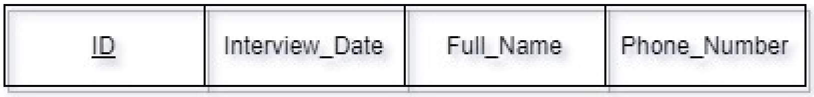
9- ROOM:



Room_Number	Extension_Number	Location	Name

The ROOM entity contains simple attributes only, so we simply include in this relation the four attributes: Name, Location, Extension_Number, and **Room_Number** which is underlined because it is a primary key.

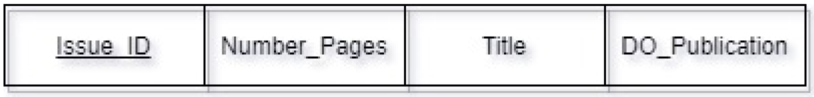
10- GUEST:



ID	Interview_Date	Full_Name	Phone_Number

The GUEST entity contains simple attributes only, so we simply include in this relation the four attributes: Full_Name, Interview_Date, Phone_Number, and **ID** which is underlined because it is a primary key.

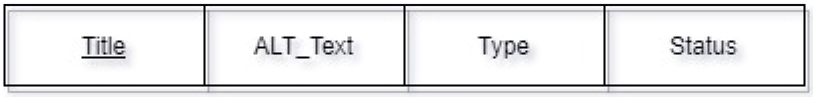
11- JOURNAL:



Issue_ID	Number_Pages	Title	DO_Publication

The JOURNAL entity contains simple attributes only, so we simply include in this relation the four attributes: Number_Pages, Title, DO_Publication, and **Issue_ID** which is underlined because it is a primary key.

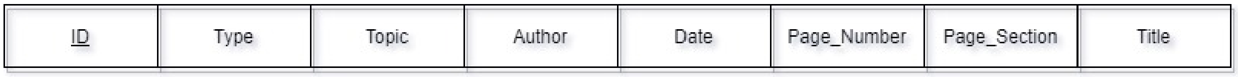
12- MEDIA:



Title	ALT_Text	Type	Status

The MEDIA entity contains simple attributes only, so we simply include in this relation the four attributes: Type, Status, ALT_Text, and **Title** which is underlined because it is a primary key.

13- ARTICLE:



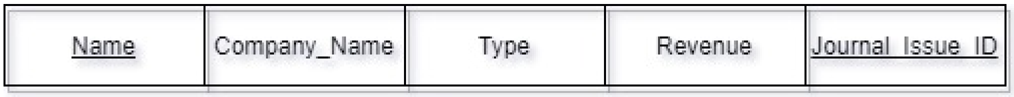
ID	Type	Topic	Author	Date	Page_Number	Page_Section	Title

The ARTICLE entity contains simple attributes only, so we simply include in this relation the six attributes: Topic, Type, Author, Date, Title and **ID** which is underlined because it is a primary key. The ARTICLE entity has *Page* as a composite attribute of which only the simple attributes *Page_Number* and *Page_Section* are included in the relation.

STEP 2: Mapping of Weak Entity Types

In this step, the weak entity types are mapped into relations. As in Step 1, only the simple attributes are included in the relations and not multivalued or derived attributes. Furthermore, weak entity relation has a foreign key attribute which is the primary key of the owner entity type. The combination of the foreign key added and the partial key of the weak entity type represent the primary key of the relation. The only weak entity in our database design is ADVERTISEMENT.

1- ADVERTISEMENT:



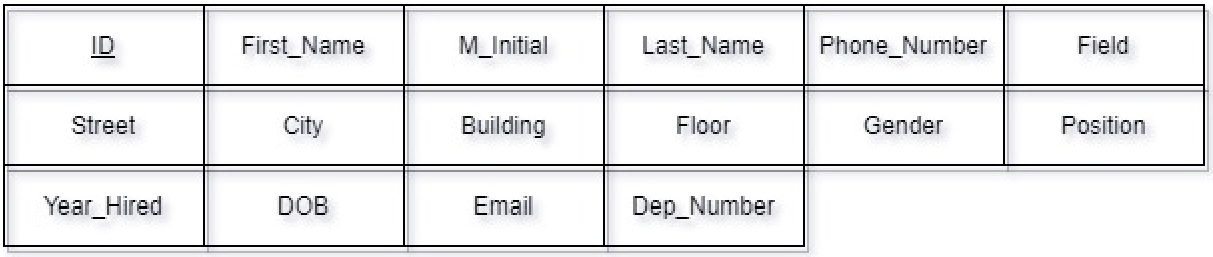
Name	Company_Name	Type	Revenue	Journal_Issue_ID

The weak entity ADVERTISEMENT does not have any derived or multivalued attributes. The simple attributes: Company_Name, Type, Revenue, and the partial key Name (underlined for that reason) are included. Moreover, the *Journal_Issue_ID*, the primary key of the owner entity JOURNAL, is included. The partial key *Name* and *Journal_Issue_ID* are combined to represent the primary key of this relation.

Step 3: Mapping of Binary 1:1 Relationship Types

To accomplish our goal, we can follow one of three approaches. The first approach, called foreign key approach is where we choose the entity on the total participation side of the relation, then we add as a foreign key the primary key of the other entity participating in this relation. The second approach, called merged relation approach is where we merge the two entities participating in the relationship into a single relation. This is only used when both participations are total and thus not useful in our case. The third approach, called cross-reference or relationship relation approach is where we create a third relation which will include the primary keys of both entities participating in the relationship. We are going to follow the foreign key approach because in our context, we will be picking out the “objects” as our foreign key, as contextually and logically speaking, this makes the most sense. (i.e., we care more about where each employee works than what employee each department contains). The binary one-to-one relationships that need to be mapped are WORKS_FOR and SUBSCRIBES_TO.

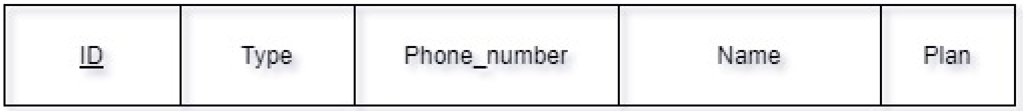
1- Employee (WORKS_FOR)



ID	First_Name	M_Initial	Last_Name	Phone_Number	Field
Street	City	Building	Floor	Gender	Position
Year_Hired	DOB	Email	Dep_Number		

Every employee works for a department. The “WORKS_FOR” relationship links the EMPLOYEE entity and the DEPARTMENT entity. On both sides of the participating entities, we have total participation. Thus, it does not matter where we add the foreign key that relates both entities; we have total participation on both sides. We chose the EMPLOYEE entity and added as a foreign key the primary key *Dep_Number* (renamed) from the DEPARTMENT entity.

2- Customer (SUBSCRIBES_TO)



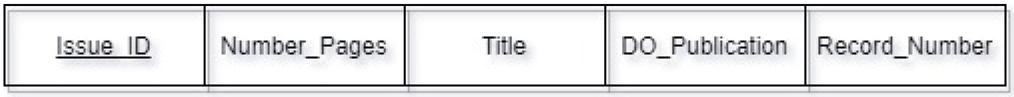
ID	Type	Phone_number	Name	Plan

Every customer subscribes to a plan. The “SUBSCRIBES_TO” relationship links the CUSTOMER entity to the PLAN entity. Every customer subscribes to a plan (or they aren’t a customer), and every plan is subscribed to. Thus, it does not matter where we add the foreign key that relates both entities; we have total participation on both sides. We chose the CUSTOMER entity and added as a foreign key the reference number, simply renamed to *Plan*.

Step 4: Mapping of Binary 1: N Relationship Types

In this step, we are going to map the binary one-to-many relationships. We add a foreign key in the entity type at the many sides of the relationship. This foreign key is the primary key of the other entity type participating in this relationship. We must also include any other simple attribute of the one-to-many relationship. The one-to-many relationships that need to be mapped are: HAS1, HAS2, HAS3, HAS4, HAS5, HAS6, HAS7, CONTACTS1, CONTACTS2, and CONTACTS3.

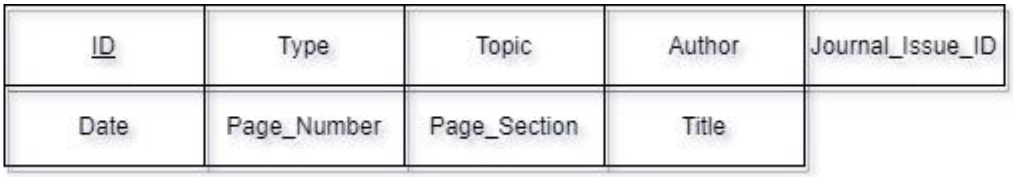
1- JOURNAL (HAS1)



Issue_ID	Number_Pages	Title	DO_Publication	Record_Number

Many journals belong to a record. The "HAS1" relationship links the JOURNAL entity and the RECORD entity. The JOURNAL entity is on the "many" side. Thus, we add to its relation the foreign key *Record_Number* which is the renamed primary key of the RECORD entity originally called *Volume_Number*.

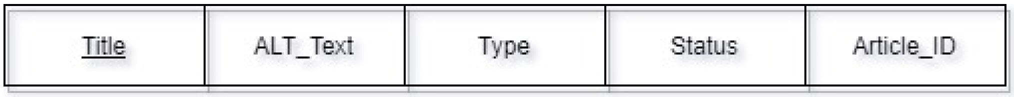
2- ARTICLE (HAS2)



ID	Type	Topic	Author	Journal_Issue_ID
Date	Page_Number	Page_Section	Title	

Many articles belong to a journal. The "HAS2" relationship links the JOURNAL entity and the ARTICLE entity. The ARTICLE entity is on the "many" side. Thus, we add to its relation the foreign key *Journal_Issue_ID* which is the renamed primary key of the JOURNAL entity originally called *Issue_ID*.

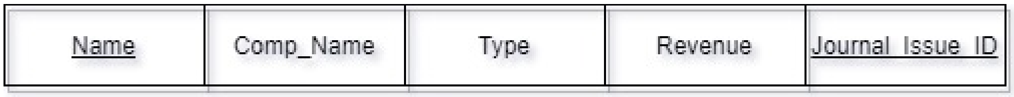
3- MEDIA (HAS3)



Title	ALT_Text	Type	Status	Article_ID

Many pieces of media have an article they are used in. The "HAS3" relationship links the MEDIA entity and the ARTICLE entity. The MEDIA entity is on the "many" side. Thus, we add to its relation the foreign key *Article_ID* which is the renamed primary key of the ARTICLE entity originally called *ID*.

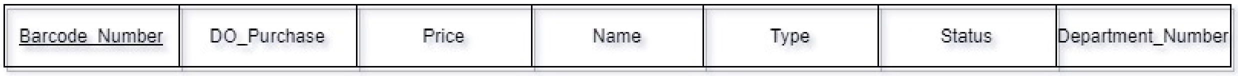
4- ADVERTISEMENT (HAS4)



Name	Comp_Name	Type	Revenue	Journal_Issue_ID

Every journal has many advertisements. The "HAS4" relationship links the ADVERTISEMENT entity and the JOURNAL entity. The ADVERTISEMENT entity is on the "many" side. Thus we add to its relation the foreign key *Journal_Issue_ID* which is the renamed primary key of the JOURNAL entity originally called *Issue_ID*. But since the ADVERTISEMENT relation schema already has the Journal_Issue_ID, nothing changes in this relation.

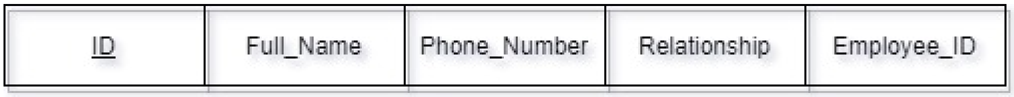
5- EQUIPMENT (HAS5)



Barcode_Number	DO_Purchase	Price	Name	Type	Status	Department_Number

Each piece of equipment belongs to a department. The "HAS5" relationship links the EQUIPMENT entity and the DEPARTMENT entity. The EQUIPMENT entity is on the "many" side. Thus we add to the relation the foreign key *Department_Number* which is the renamed primary key of the DEPARTMENT entity originally called *Number*.

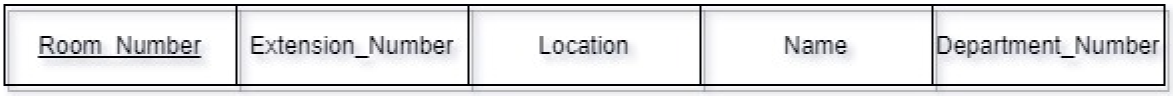
6- DEPENDENT (HAS6)



ID	Full_Name	Phone_Number	Relationship	Employee_ID

An employee may depend on many people. The “HAS6” relationship links the DEPENDENT entity and the EMPLOYEE entity. The DEPENDENT entity is on the “many” side. Thus, we add to its relation the foreign key *Employee_ID* which is the renamed primary key of the EMPLOYEE entity originally called *ID*.

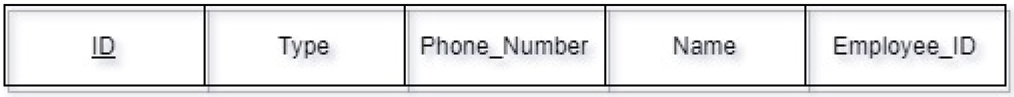
7- ROOM (HAS7)



Room_Number	Extension_Number	Location	Name	Department_Number

Every room belongs to one department. The “HAS7” relationship links the ROOM entity and the DEPARTEMENT entity. The ROOM entity is on the “many” side. Thus, we add to its relation the foreign key *Number* which is the primary key of the DEPARTMENT entity, and we rename it *Department_Number*.

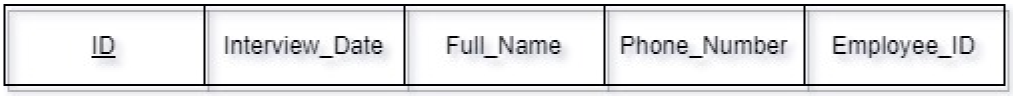
8- CUSTOMER (CONTACTS1)



ID	Type	Phone_Number	Name	Employee_ID

Every customer is contacted by an employee. The “CONTACTS1” relationship links the CUSTOMER entity and the EMPLOYEE entity. The CUSTOMER entity is on the “many” side. Thus we add to its relation the foreign key *Employee_ID* which is the renamed primary key of the EMPLOYEE entity originally called *ID*.

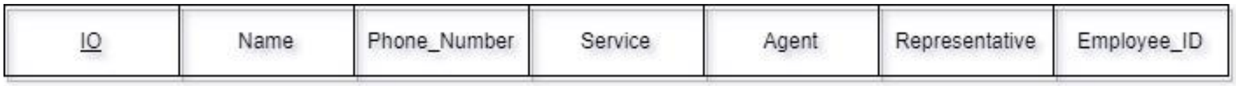
9- GUEST (CONTACTS2)



ID	Interview_Date	Full_Name	Phone_Number	Employee_ID

Every guest is contacted by an employee. The “CONTACTS2” relationship links the GUEST entity and the EMPLOYEE entity. The GUEST entity is on the “many” side. Thus we add to its relation the foreign key *Employee_ID* which is the renamed primary key of the EMPLOYEE entity originally called *ID*.

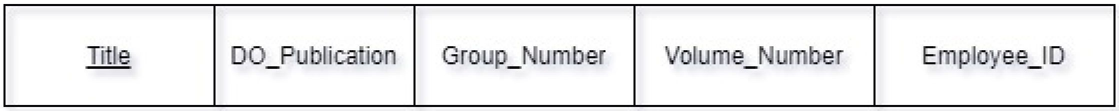
10- AGENCY (CONTACTS3)



IO	Name	Phone_Number	Service	Agent	Representative	Employee_ID

Every agency is contacted by an employee. The “CONTACTS3” relationship links the AGENCY entity and the EMPLOYEE entity. The AGENCY entity is on the “many” side. Thus we add to its relation the foreign key *Employee_ID* which is the renamed primary key of the EMPLOYEE entity originally called *ID*.

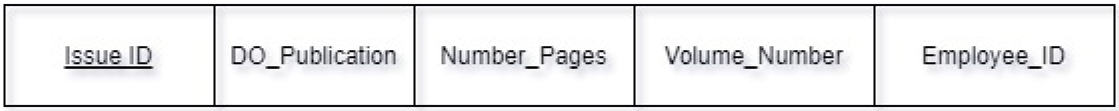
11- RECORD (MANAGES1)



Title	DO_Publication	Group_Number	Volume_Number	Employee_ID

Every record is managed by an employee. The “MANAGES” relationship links the RECORD entity and the EMPLOYEE entity. The RECORD entity is on the “many” side. Thus, we add to its relation the foreign key *Employee_ID* which is the renamed primary key of the EMPLOYEE entity originally called *ID*.

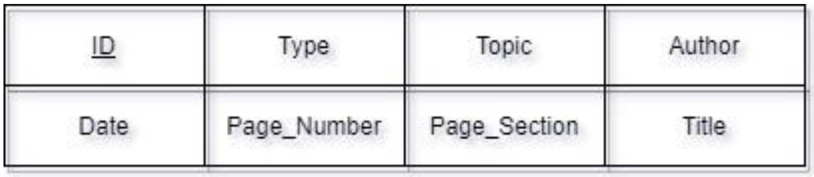
12- JOURNAL (MANAGES2)



Issue ID	DO_Publication	Number_Pages	Volume_Number	Employee_ID

Every journal is managed by an employee. The “MANAGES” relationship links the JOURNAL entity and the EMPLOYEE entity. The JOURNAL entity is on the “many” side. Thus we add to its relation the foreign key *Employee_ID* which is the renamed primary of key of the EMPLOYEE entity originally called *ID*.

13- ARTICLE (WRITES)

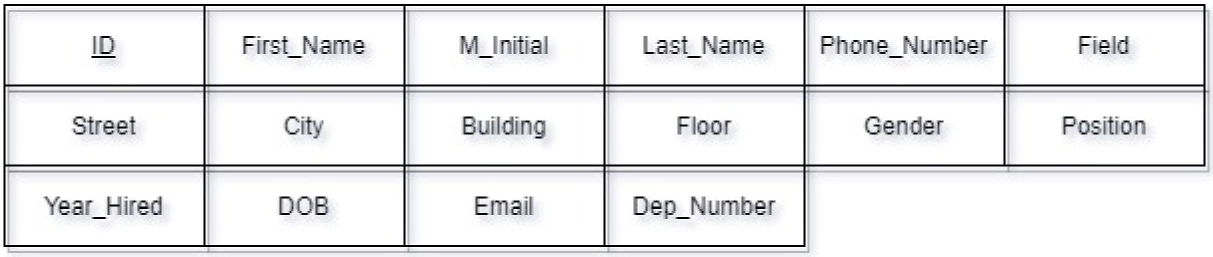


ID	Type	Topic	Author
Date	Page_Number	Page_Section	Title

Many articles are written by an employee. The “WRITES” relationship links the ARTICLE entity and the EMPLOYEE entity. The ARTICLE entity is on the “many” side. Thus, we add to its relation the foreign key *Author* which is the renamed primary key of the EMPLOYEE entity originally called *ID*. But since the ARTICLE relation schema already has the *Author* attribute, nothing changes in this relation.

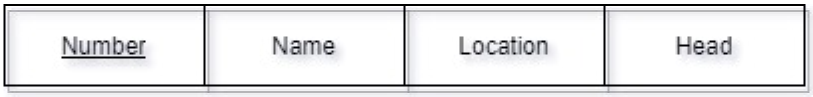
FINAL STEP: Final Displays:

EMPLOYEE:



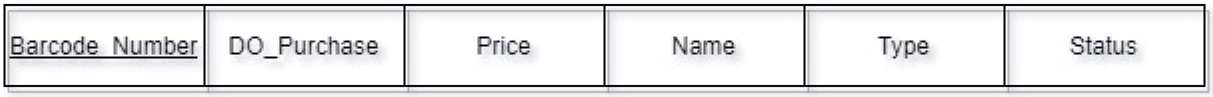
ID	First_Name	M_Initial	Last_Name	Phone_Number	Field
Street	City	Building	Floor	Gender	Position
Year_Hired	DOB	Email	Dep_Number		

DEPARTMENT:



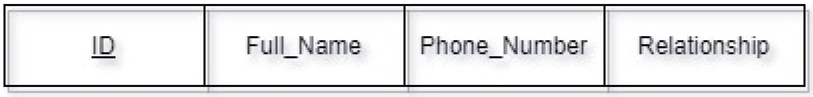
Number	Name	Location	Head

EQUIPMENT



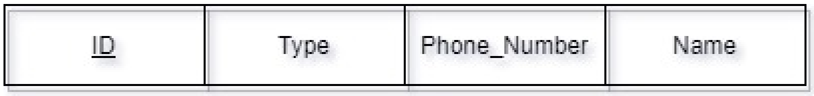
Barcode_Number	DO_Purchase	Price	Name	Type	Status

DEPENDENT:



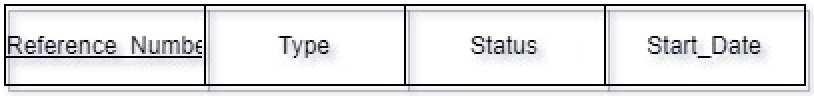
ID	Full_Name	Phone_Number	Relationship

CUSTOMER:



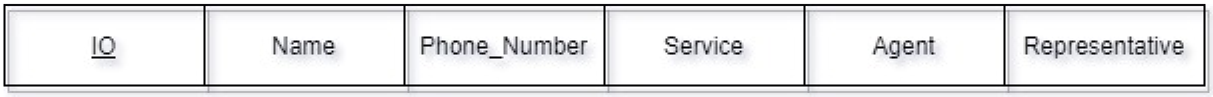
ID	Type	Phone_Number	Name

PLAN:



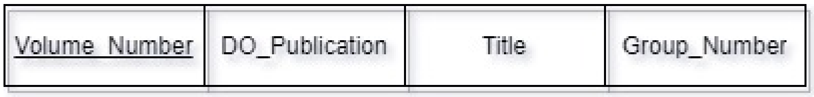
Reference_Numbe	Type	Status	Start_Date

AGENCY:



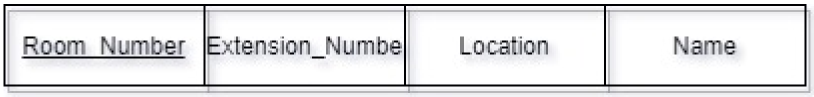
IO	Name	Phone_Number	Service	Agent	Representative

RECORD:



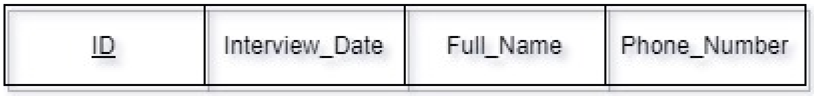
Volume_Number	DO_Publication	Title	Group_Number

ROOM:



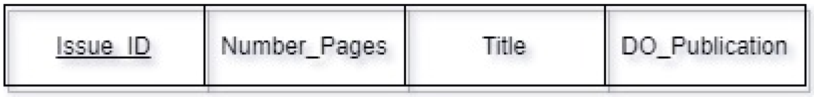
Room_Number	Extension_Numbe	Location	Name

GUEST:



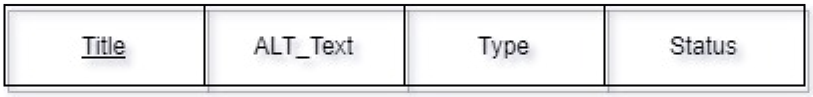
ID	Interview_Date	Full_Name	Phone_Number

JOURNAL:



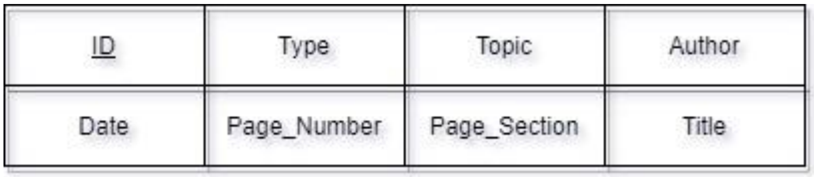
Issue_ID	Number_Pages	Title	DO_Publication

MEDIA:



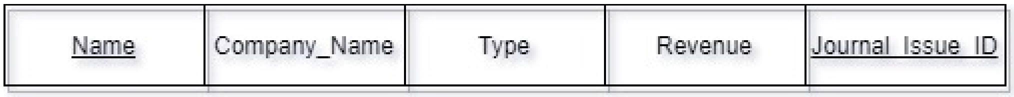
Title	ALT_Text	Type	Status

ARTICLE:



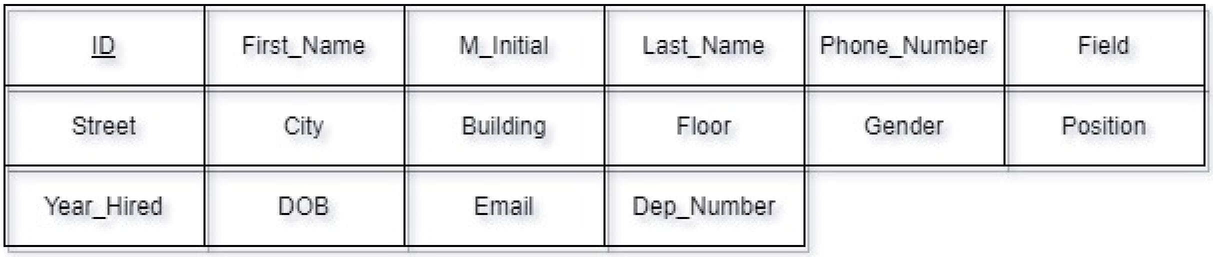
ID	Type	Topic	Author
Date	Page_Number	Page_Section	Title

ADVERTISEMENT:



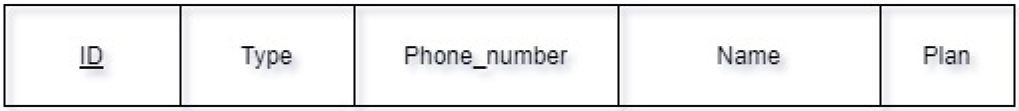
Name	Company_Name	Type	Revenue	Journal_Issue_ID

WORKS_FOR



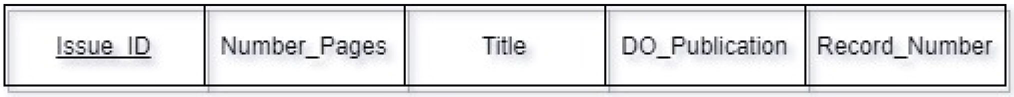
ID	First_Name	M_Initial	Last_Name	Phone_Number	Field
Street	City	Building	Floor	Gender	Position
Year_Hired	DOB	Email	Dep_Number		

SUBSCRIBES_TO



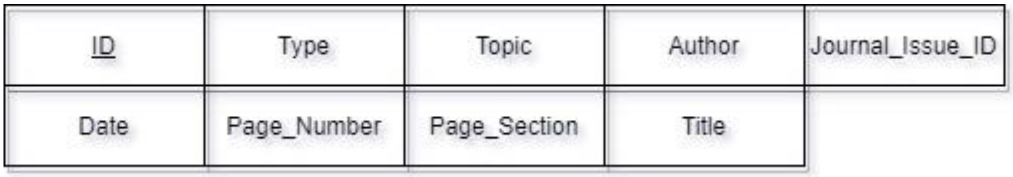
ID	Type	Phone_number	Name	Plan

HAS1



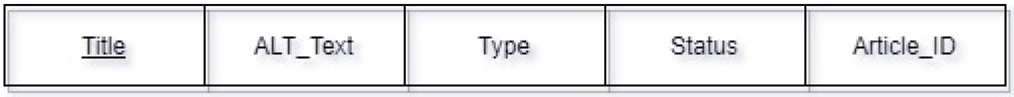
Issue_ID	Number_Pages	Title	DO_Publication	Record_Number

HAS2



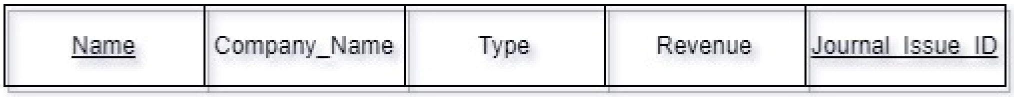
ID	Type	Topic	Author	Journal_Issue_ID
Date	Page_Number	Page_Section	Title	

HAS3



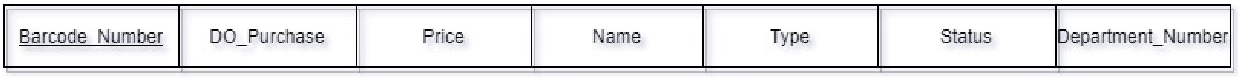
Title	ALT_Text	Type	Status	Article_ID

HAS4



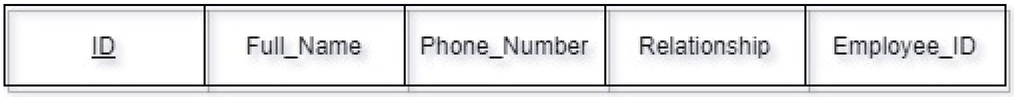
Name	Company_Name	Type	Revenue	Journal_Issue_ID

HAS5



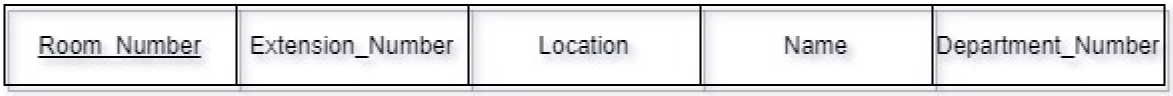
Barcode_Number	DO_Purchase	Price	Name	Type	Status	Department_Number

HAS6



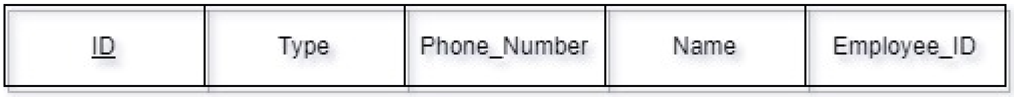
ID	Full_Name	Phone_Number	Relationship	Employee_ID

HAS7



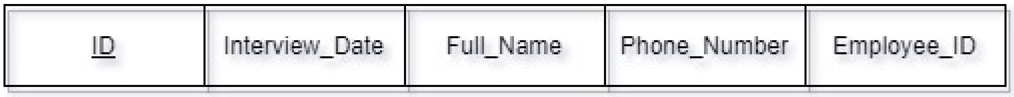
Room_Number	Extension_Number	Location	Name	Department_Number

CONTACTS1



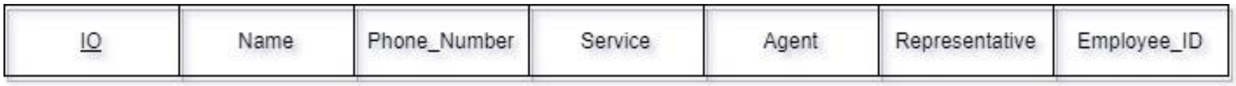
ID	Type	Phone_Number	Name	Employee_ID

CONTACTS2



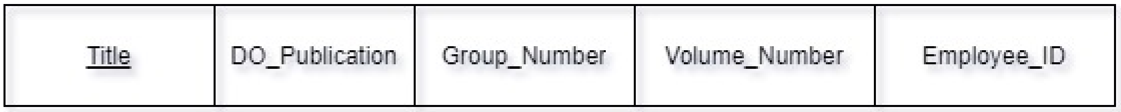
ID	Interview_Date	Full_Name	Phone_Number	Employee_ID

CONTACTS3



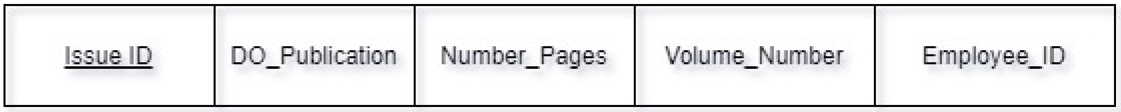
IO	Name	Phone_Number	Service	Agent	Representative	Employee_ID

MANAGES1



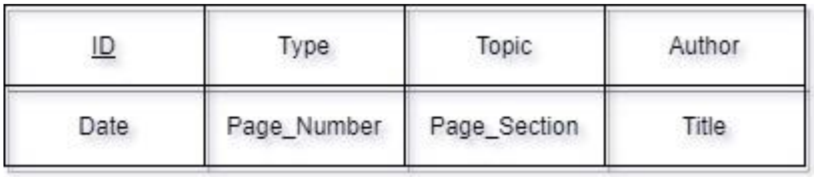
Title	DO_Publication	Group_Number	Volume_Number	Employee_ID

MANAGES2



Issue ID	DO_Publication	Number_Pages	Volume_Number	Employee_ID

WRITES



ID	Type	Topic	Author
Date	Page_Number	Page_Section	Title

Conclusion:

For an organization as complex and fast-paced as a newspaper agency, it is crucial for a database to be functioning reliably and efficiently to keep track of the hundreds of employees, transactions and journals that are in circulation every day. Our database model effectively gives an organization the structure it needs to maintain this reliability and efficiency while considering many factors that could be overlooked. Specifically, we believe our model covers all the necessities as well as provides the finer details. The system is life-like, having been inspired by an actual newspaper organization. But like all things, there is much room for improvement that we are sure Professor Ramzi Haraty will provide in detail. Overall, we believe our system and its organization has its strengths and could prove a viable solution for organizations everywhere.

Instructor Comments: