**Lebanese American University**

**School of Arts and Sciences**

**Database Management Systems**

**CSC 375**

****

**Final Project Report**

**Bahaa Thebian (201904844) - Charbel Younis (201803601)**

**Christopher Chalhoub (201804127) - George Freiha (201806517)**

**Hassan Abed Ali (201905280) - Moussa Haidous (201902042)**

**11/23/20**

[**I- Introduction**](#_wah3x8m29ghl) **3**

[**II- ER Diagram**](#_2tp9ykb4dja1) **4**

[**III- Entities**](#_z3nvld11jljx) **5**

[Space:](#_31nydntm66au) 6

[Client:](#_qveyf66x235u) 7

[Point\_of\_Contact:](#_oya1xsar9j71) 8

[Rent\_Contract:](#_4grlcdxq4ypc) 9

[Department:](#_w0yk2urpm4j6) 9

[Employee:](#_kj9dpiyomgym) 10

[Employment\_Contract:](#_fmc3yl7s3p3k) 11

[Floor:](#_2zz074y7920j) 12

[Event:](#_x2yfby5ej5r2) 13

[Music:](#_nmofw1g4oiii) 13

[Parking\_Lot:](#_x88en6sj73d8) 14

[**IV- Relationships:**](#_r9bs39yswdlc) **15**

[HAS:](#_7h417cag19js) 15

[SIGNS\_1:](#_jq542ro37ltx) 15

[WORKS\_AT:](#_1bpjjtrj1cw6) 16

[MANAGES:](#_h0plly4kofdw) 16

[SIGNS\_2:](#_cd7r4g72wrqu) 17

[CONTAINS\_1:](#_6q1hkh502s4j) 17

[CONTAINS\_2:](#_bxbr6au081ha) 18

[Plays:](#_9mwvdnlijaj0) 18

[CONTAINS\_3:](#_f45jttkyknl0) 19

[HAS\_2:](#_7c3mxo78g7ma) 19

[CONTAINS\_4:](#_b0dc0bjcguhs) 20

[**V- ER to Relations Mapping**](#_wh1g71c6sb57) **21**

[Step 1: Mapping of Regular Entity Types](#_ldifckfq3xi8) 21

[Step 2: Mapping of Weak Entity Types](#_cwz9npzifi7a) 25

[Step 3: Mapping of Binary 1:1 Relationship Types](#_p8hibot2rxyd) 26

[Step 5: Mapping of M:N Relationship Types](#_j6iq652rmic2) 31

[Step 6: Mapping of Multivalued Attributes](#_uv5qhg7rhnvf) 32

[Final Displays:](#_i5n0jcd8v6z5) 33

[**VI - Database Creation**](#_4aupche8zkp3) **36**

[Step 1: Table Creation](#_g2qg85aqh754) 36

[Step 2: Data Insertion](#_a9p4rrclfez) 52

[Final Display:](#_dz6tkvltu8z) 81

[**Sample Transactions**](#_3c0shcw2wxo) **94**

[**VII- Normalization Up to The BCNF Normal Form:**](#_deoqwfbvlre5) **107**

[First Normal Form:](#_8ethy2gz702p) 108

[Second Normal Form:](#_esrccfhyuzn) 108

[Third Normal Form:](#_r99n2p3un5gr) 109

[Boyce Codd Normal Form:](#_j626cspjvzem) 109

[**VIII - Conclusion**](#_fsufk3ptutmk) **120**

# 

# 

# 

# 

# 

# 

# 

# 

# 

# 

# 

# 

# 

# 

# 

# 

# I- Introduction

In our day and age, shopping cannot pose an inconvenience. Say you want to buy some clothes, then maybe buy yourself a nice pair of headphones to listen to music. You’ll have to plan your day around visiting multiple shops and consider the commute to these multiple shops. You might not even get it all done in one day and if you do, you’ll reach home exhausted. What a waste of our most valuable resource, time!

We provide a multi-level building with tens of shops of all types and kinds such as clothing shops so that you, the customer, can get your fabulous clothes, Michelin star restaurants to cater to all your gourmet needs, and most definitely, electronic stores so that you can hear your music through the best headphones in the market. Not only that, but our floors are always playing the newest most popular music so that the customer can enjoy their shopping experience. The best part is the customer can get that all done in one visit!

Our shopping mall is known for its simplistic relationship with its client, where each client has the freedom to choose its suitable shop in terms of space required, rent payment and others. This feature is due to our great database design that handles complex transactions and makes them look simplistic.

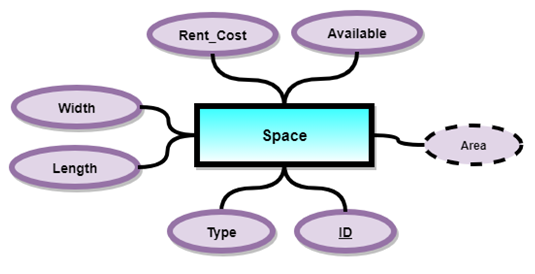
Moreover, the shopping mall management team always wants to keep an environment that is comfortable for family and friends by including seasonal and occasional events like Christmas, Halloween and other themes. Having huge free middle space areas, we made it easy for sponsors to organize any additional event needed.

Considering how dense and hard it is to find a parking spot in the city, we provide parking to not only our employees but all customers as well.

# II- ER Diagram

# III- Entities

## Space:

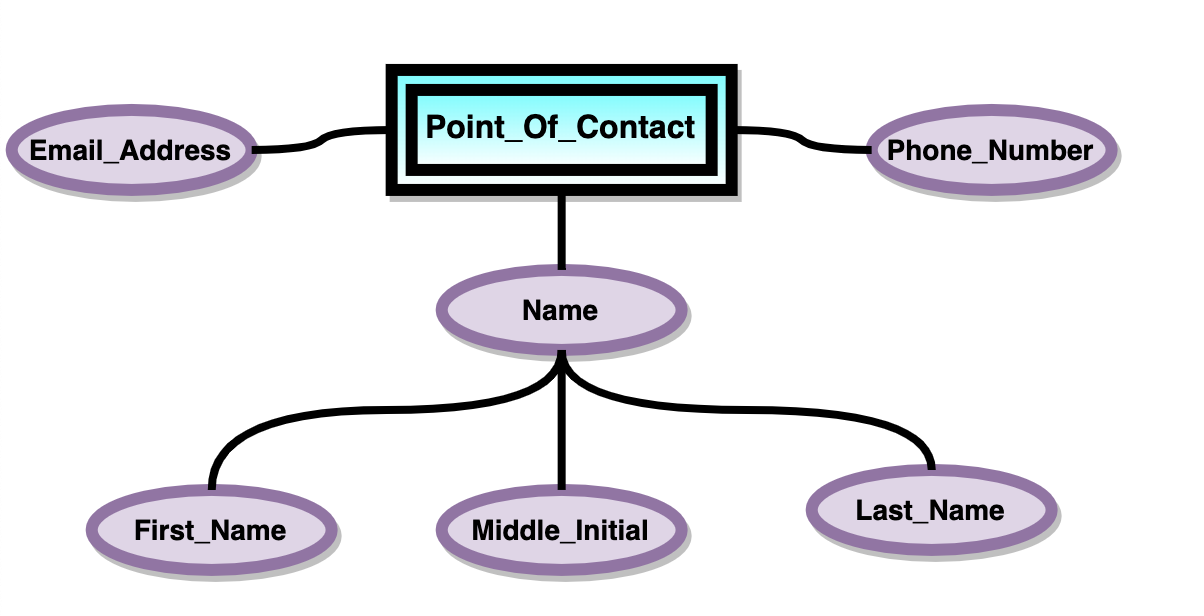


Spaces are the basis of any mall and make up the foundation of all the beautiful stores, restaurants, and advertisements that’ll comprise the mall. The space is identified by its primary key **ID**. The space is described by its **Length** and **Width** that can be used to calculate the **Area** (derived attribute). The space is also described by its **Type**. It could be the cornerstone of a Store, a Restaurant, or an Advertisement. Most importantly, the space holds information regarding if it's **Available**, and accordingly its **Rent\_Cost**.

## Client:

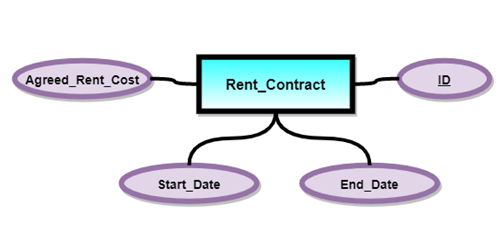
The client is of utmost importance to our mall, hence we keep track of all the information regarding them to deliver the best service. To identify our myriad clients, we use their primary key **ID**. We keep track of our client’s **Business\_Name**, **Phone\_Number**, and **Email\_Address** to be able to reach them at any time. We also have their **Address** which is a composite attribute made of **Street\_Name** and **City** to ensure they’re reachable at any time. Finally, we hold data regarding the **Rented\_space\_No** to provide them with the best possible customer service.

## Point\_of\_Contact:



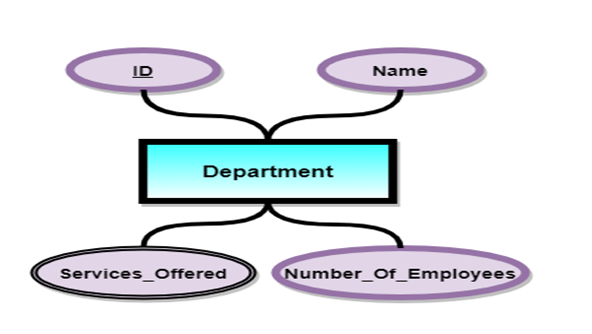
The weak entity point of contact corresponds to its parent entity **Client**. The client’s point of contact is the main person of which we do all our negotiations, deals, and contracts through. Every point of contact has a composite attribute **Name** consisting of their **First\_Name**, **Middle\_Initial**, and **Last\_Name**. Since they’re our main point of contact with our client, we also save their **Phone\_Number** and an **Email\_Address** to be the partial key of this weak entity.

## Rent\_Contract:

****

The rent contract is what brings our entire business altogether by binding our client with one of our spaces. Each rent contract is identified by its **ID** and describes the **Agreed\_Rent\_Cost** which might be different from the space’s advertised rent cost. We also keep track of the **Start\_Date** and the **End\_Date** of the contract, essentially specifying when the contract expires and is available to be renewed and/or rented by a different client.

## Department:

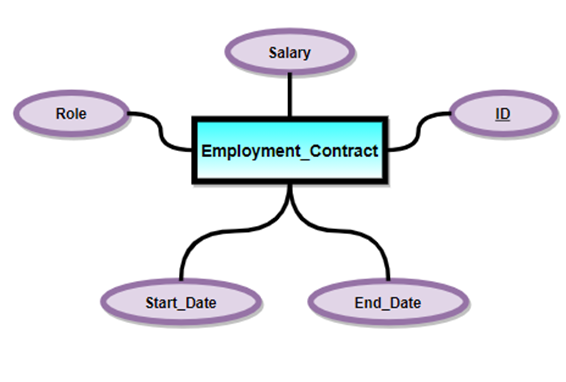


Departments are the backbone of our mall and make sure all our different aspects are up to speed and as they should be. The department is identified by its primary key **ID**. Each department has a different **Name** and a multivalued attribute **Services\_Offered** which could be financial services, customer service, public relations, and more. We keep track of the **Number\_of\_Employees** employed by each department to ensure that all our departments are appropriately serviced and of any possible job openings for our loving community.

## Employee:

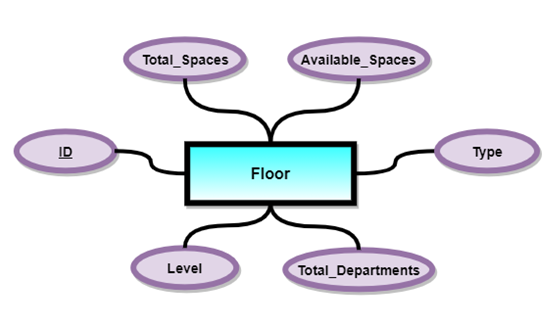
The employee makes sure everything at our mall is running as it should. Our employees are identified by their primary key **ID**. Every employee has a composite attribute **Name** consisting of their **First\_Name**, **Middle\_Initial**, and **Last\_Name**. We also maintain their **Date\_of\_Birth** and **Gender** for tax filing purposes. Moreover, we keep track of their **Email**, **Phone\_Number**, and **Address** which is a composite attribute made of **Street\_Name** and **City** to ensure they’re reachable at any time.

## Employment\_Contract:

****

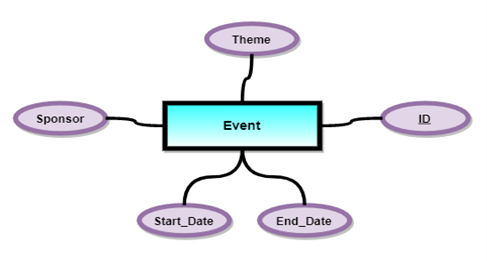
The employee contract binds our employees to our mall. Each Employment contract is identified by its **ID** and describes the **Salary** alongside the **Role** of the employee. We also keep track of the **Start\_Date** and the **End\_Date** of the contract, essentially specifying when the contract expires and is available to be renewed by the employee.

## Floor:

****

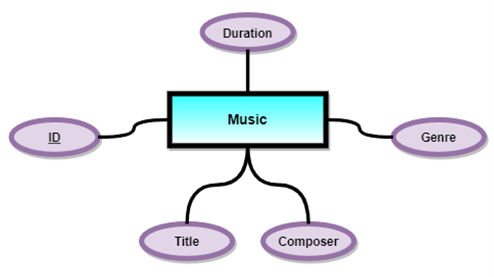
At our mall, every Floor specializes in a certain **Type** of space whether it's restaurants, clothing stores, sports equipment, or others. Hence ensuring our mall customers can easily navigate the mall. We identify each floor by its **ID** alongside the **Level** name. Moreover, we keep track of the **Total\_Departments** to ensure all our floors are serviced sufficiently. Also, we keep track of the number of **Total\_Spaces** and **Available\_Spaces** in each floor so that we can distribute stores and restaurants correctly.

## Event:



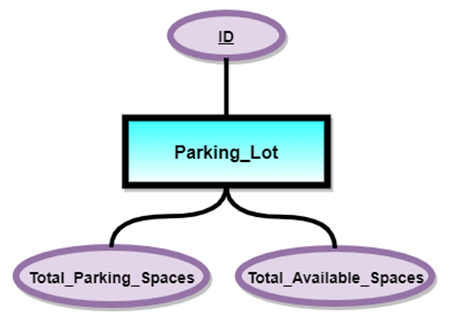
To keep our mall lively and exciting, we rotate different events across the entire year. Each event is identified by its **ID** and describes the **Theme** (such as christmas, Easter, etc.), and the event’s **Sponsor**. We also maintain the **Start\_Date** and **End\_Date** of each event so that we can schedule multiple events efficiently.

## Music:



To entertain our customers, employees and our event attendees, we purchase music frequently. Hence, we keep track of our music repository and maintain data regarding its **Title**, **Composer**, **Genre**, and **Duration** so that we can choose what music is appropriate to the event or the current mall climate. We also identify all our music by their primary key **ID.**

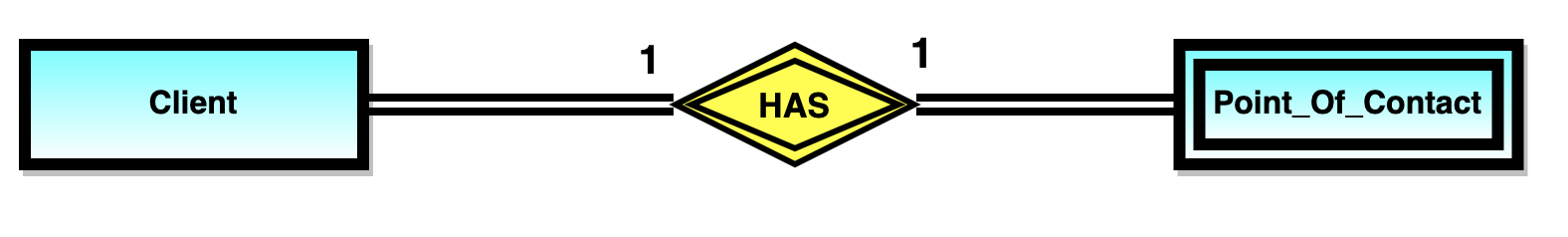
## Parking\_Lot:



We offer free parking at our mall to all our customers, thanks to the multiple parking lots we maintain. We identify each parking lot by its **ID**. Moreover, we keep track of the number of **Total\_Parking\_Spaces** and **Total\_Available\_Spaces** in each parking lot such that our customers can enter a parking lot with utmost confidence they’ll find a parking slot.

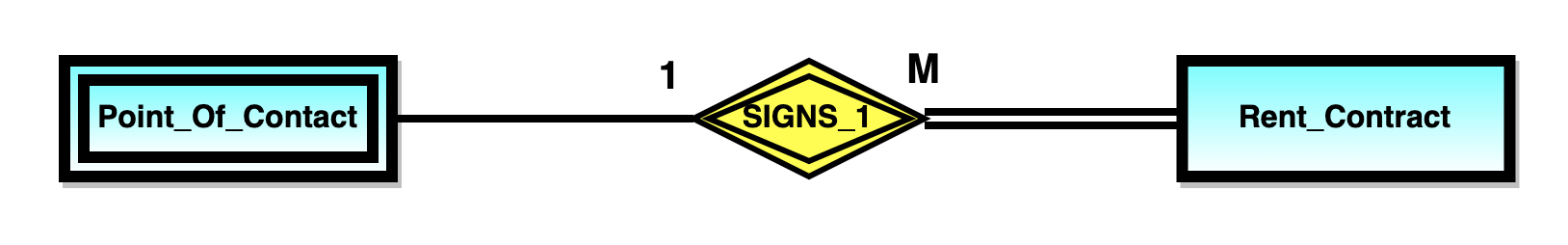
# IV- Relationships:

## HAS:



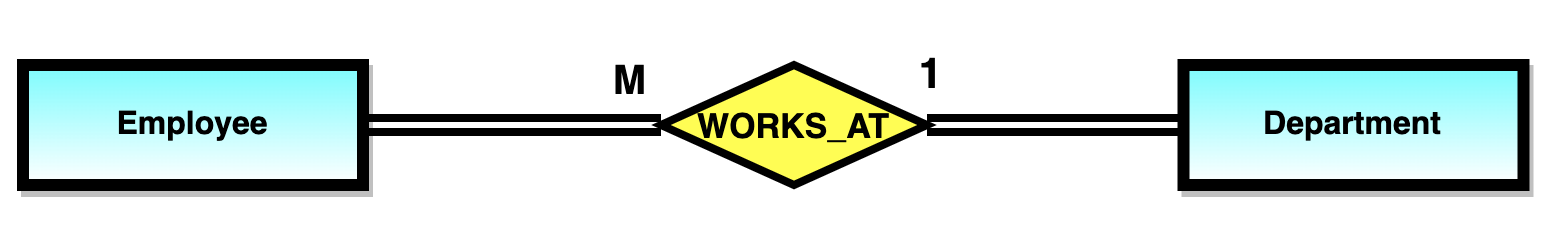
Every **CLIENT** has a **POINT\_OF\_CONTACT**. Each **POINT\_OF\_CONTACT** requires a **CLIENT** and every **CLIENT** is represented by a **POINT\_OF\_CONTACT**. Hence, a weak 1-to-1 total participation identifying ***HAS*** relation is created between every **CLIENT** and **POINT\_OF\_CONTACT**.

## SIGNS\_1:

****

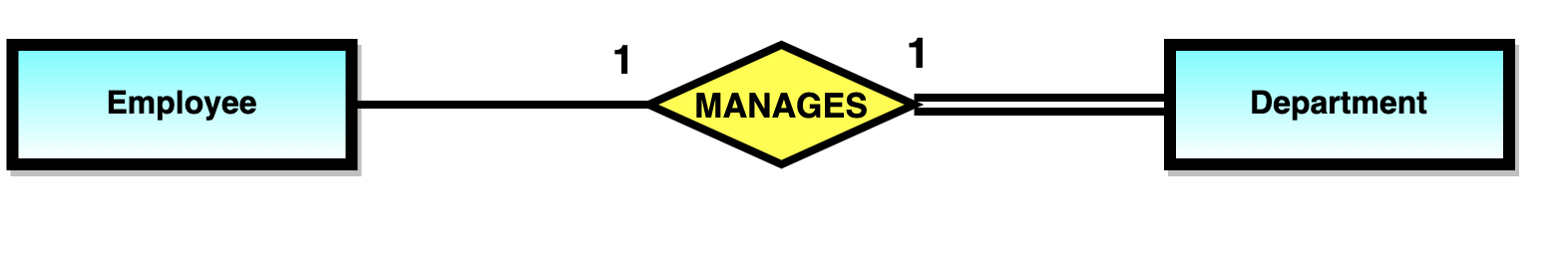
Some **POINT\_OF\_CONTACT** might sign a **RENT\_CONTRACT**(s). Not every **POINT\_OF\_CONTACT** has to sign a **RENT\_CONTRACT**(s) but every **RENT\_CONTRACT**(s) requires a **POINT\_OF\_CONTACT** to sign it. Hence, a 1-to-M partial participation from **POINT\_OF\_CONTACT** and total participation from **RENT\_CONTRACT** ***SIGNS\_1*** relation is created between some **POINT\_OF\_CONTACT** and every **RENT\_CONTRACT**.

## WORKS\_AT:

****

Every **EMPLOYEE**(s) works at a **DEPARTMENT**. Every **EMPLOYEE**(s) has to work at a **DEPARTMENT** and every **DEPARTMENT** requires an **EMPLOYEE**(s) to work at it. Hence, a M-to-1 total participation ***WORKS\_AT*** relation is created between every **EMPLOYEE**(s)and every **DEPARTMENT**.

## MANAGES:

****

Some **EMPLOYEE** manages a **DEPARTMENT**. Not every **EMPLOYEE** manages a **DEPARTMENT** but every **DEPARTMENT** requires an **EMPLOYEE**(s) to manage it. Hence, a 1-to-1 partial participation from **EMPLOYEE** and total participation from **DEPARTMENT** ***Manages*** relation is created between some **EMPLOYEE** and every **DEPARTMENT**.

## SIGNS\_2:

****

Every **EMPLOYEE** signs an **EMPLOYMENT\_CONTRACT**. Every **EMPLOYEE** has to sign an **EMPLOYMENT\_CONTRACT** and every **EMPLOYMENT\_CONTRACT** requires an **EMPLOYEE** to sign it. Hence, a 1-to-1 total participation ***WORKS\_AT*** relation is created between every **EMPLOYEE**(s)and every **DEPARTMENT**.

## CONTAINS\_1:

****

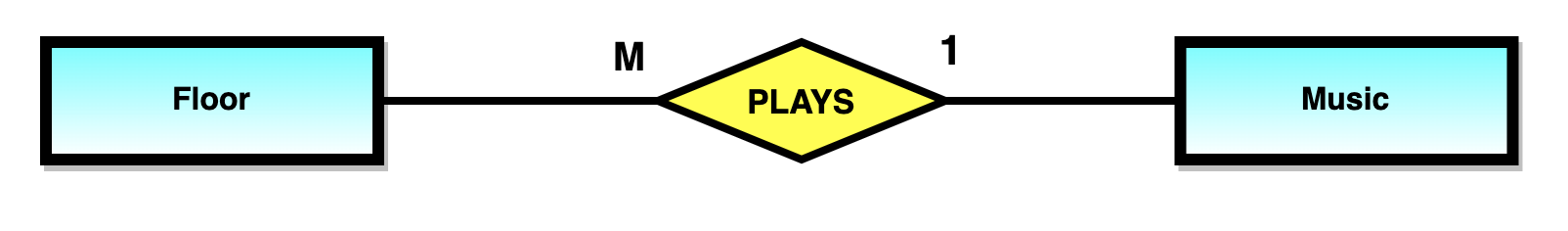
Some **FLOOR** might contain a **DEPARTMENT**(s). Not every **FLOOR** has to contain a **DEPARTMENT**(s) but every **DEPARTMENT**(s) exists on some **FLOOR**. Hence, a 1-to-M partial participation from **FLOOR** and total participation from **DEPARTMENT**(s) ***CONTAINS\_1*** relation is created between some **FLOOR** and every **DEPARTMENT**(s).

## CONTAINS\_2:

****

Some **FLOOR** might contain a **SPACES**(s). Not every **FLOOR** has to contain a **SPACE**(s) but every **SPACE**(s) exists on some **FLOOR**. Hence, a 1-to-M partial participation from **FLOOR** and total participation from **SPACE**(s) ***CONTAINS\_2*** relation is created between some **FLOOR** and every **SPACE**(s).

## Plays:

****

Some **FLOOR**(s) might play **MUSIC**. Not every **FLOOR**(s) has to play **MUSIC** nor does every **MUSIC** has to be played on some **FLOOR**. Hence, a M-to-1 partial participation ***PLAYS*** relation is created between some **FLOOR**(s)and some **MUSIC**.

## CONTAINS\_3:

****

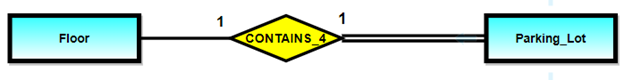
Some **FLOOR**(s) might contain an **EVENT**(s). Not every **FLOOR**(s) has to contain an **EVENT**(s) but every **EVENT**(s) requires a **FLOOR**(s). Hence, a M-to-M partial participation from **FLOOR** and total participation from **EVENT**(s) ***CONTAINS\_3*** relation is created between some **FLOOR**(s)and every **EVENT**(s).

## HAS\_2:

****

Every **RENT\_CONTRACT must be bound to only one SPACE**. Not All **SPACES** have to be bound to a **RENT\_CONTRACT** since some of them might not be rented yet. Hence, a 1-to-1 partial participation ***HAS\_2*** relation is created between **SPACE** and **RENT\_CONTRACT**.

## CONTAINS\_4:

****

Every **PARKING\_LOT** must exist on a **FLOOR**. Not All **Floors** have to contain a **PARKING\_LOT** . Hence, a 1-to-1 partial participation ***CONTAINS\_4*** relation is created between **FLOOR** and **PARKING\_LOT** .

# 

# V- ER to Relations Mapping

## Step 1: Mapping of Regular Entity Types

The first step should include the mapping from regular entity types into relations. Every regular entity will have its own relation which includes all the simple attributes, and a single primary key that is underlined( and orange colored). The regular (strong) entities for the Shopping Mall are:

* **Employee**
* **Employment\_Contract**
* **Department**
* **Space**
* **Rent\_Contract**
* **Client**
* **Music**
* **Event**
* **Floor**
* **Parking\_Lot**

**1.**  **Employee:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **ID** | **First\_Name** | **Middle\_Initial** | **Last\_Name** | **Date\_of\_Birth** |
| **Gender** | **Email** | **Phone number** | **City** | **Street\_Name** |

The **EMPLOYEE** entity contains simple and composite attributes. The composite attributes in the Employee entity are Name which has three simple attributes **First\_Name**, **Middle\_Initial**, and **Last\_Name**. and Address which has two simple attributes that are **City** and **Street\_Name**. Additionally, the simple attributes of this entity are **Phone\_Number**, **Email**, **Gender**, **Date\_Of\_Birth**, and the primary key **ID** which is underlined.

**2.** **Employment Contract:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **ID** | **Start\_Date** | **End\_Date** | **Salary** | **Role** |

The **EMPLOYEMENT\_CONTRACT** entity contains only simple attributes, so we directly include in this relation the attributes **Start\_Date**, **End\_Date**, **Salary**, **Role**, and the primary key **ID** which is underlined.

**3.** **Department:**

|  |  |  |
| --- | --- | --- |
| **ID** | **Name** | **NbEmployees** |

The **DEPARTMENT** entity contains simple and multivalued attributes. The multivalued attribute Service\_Offered is not represented in this relation. Furthermore, the simple attributes in this relation are **Name**, **Number of employees**, and primary key **ID** which is underlined.

**4.** **Space:**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **ID** | **Length** | **Width** | **Available** | **Rent\_Cost** | **Type** |

The **SPACE** entity contains simple and derived attributes. The derived attribute **Area** is not included. Furthermore, the simple attributes in this relation are **Length**, **Width**, **Available**, **Rent\_Cost**, **Type** and the primary key **ID** which is underlined.

**5.** **Rent Contract:**

|  |  |  |  |
| --- | --- | --- | --- |
| **ID** | **Start\_Date** | **End\_Date** | **Rent\_Cost** |

The **RENT\_CONTRACT** entity contains only simple attributes so we directly include in this relation the attributes **Start\_Date**, **End\_Date**, **Rent\_Cost**, and the primary key **ID** which is underlined.

**6. Event:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **ID** | **Name** | **Sponsor** | **Start Date** | **End Date** |

The **EVENT** entity contains only simple attributes, so we directly include in this relation the attributes **Name**, **Sponsor**, **Start\_Date**, **End\_Date**, and the primary key **ID** which is underlined.

**7. Music:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **ID** | **Title** | **Composer** | **Genre** | **Duration** |

The **MUSIC** entity contains only simple attributes, so we directly include in this relation the attributes **Title**, **Composer**, **Genre**, **Duration**, and the primary key **ID** which is underlined.

**8.** **Client:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **ID** | **Business Name** | **Phone Number** | **Email Address** | **Rented\_Spaces\_Nb** |
| **City** | **Street\_Name** |  | | |

The **CLIENT** entity contains simple and composite attributes. The CLIENT entity has Address as a composite attribute divided into 2 simple attributes **City** and **Street\_Name** represented in this relation. Moreover, this relation includes simple attributes which are **Business\_Name**, **Phone\_Number**, **Email\_Address**, **Rented\_Spaces\_Number**, and the primary key **ID** which is underlined.

**9.** **Floor:**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **ID** | **Level** | **Total spaces** | **Available Spaces** | **Total Departments** | **Type** |

The **FLOOR** entity contains only simple. The simple attributes in this relation are **Level**, **Total\_Spaces**, **Available\_Spaces**, **Total\_Departments**, **Type** and the primary key **ID** which is underlined.

**10. Parking Lot:**

|  |  |  |
| --- | --- | --- |
| **ID** | **Total parking spaces** | **Available spaces** |

The **PARKING\_LOT** entity contains only simple attributes, so we directly include in this relation the attributes **Total\_Parking\_Spaces**, **Available\_Spaces**, and the primary key **ID** which is underlined.

## Step 2: Mapping of Weak Entity Types

The second step should include the mapping of weak entity types into relations without including the multivalued or derived attributes. Moreover, the weak entity relation has a foreign key attribute which is the primary key of a specific 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. There is only one weak entity for the Shopping mall which is:

* **POINT\_OF\_CONTACT**.

**1. Point\_of \_Contact:**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Email** | **Client\_ID** | **First\_Name** | **Middle\_Initial** | **Last\_Name** | **Phone\_Number** |

The weak entity **POINT\_OF\_CONTACT** does not include any multivalued nor derived attributes. The composite attribute in the **POINT\_OF\_CONTACT** entity is Name which has three simple attributes **First\_Name**, **Middle\_Name**, and **Last\_Name**. Moreover, the simple attributes of this entity are **Phone\_Number**, **Email\_Address**. The **Client\_ID** is also included, it is the primary key of the client entity type. The combination of both the partial keys **Email**, and foreign key **Client\_ID** represents the primary key of this relation.

## Step 3: Mapping of Binary 1:1 Relationship Types

In this step, we are going to map the binary one-to-one relationships. In order 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 a 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 it is the most useful in our case. The binary one-to-one relationships that need to be mapped are:

* **Manages**
* **Signs\_2**
* **Has**
* **Has\_2**

**1.** **MANAGES (Department schema):**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **ID** | **Manager\_ID** | **Manager \_experience** | **Name** | **NbEmployees** |

Every **DEPARTMENT** requires a Manager, Thus “Manages” relationship links the **EMPLOYEE** entity and the **DEPARTMENT** entity. Not all Employees are Managers, so the participation is partial from the employee side. However, Every Department Must have a Manager which shows Total participation from the Department side. Hence, we chose the Department relation in which we added, as a foreign key, the primary key ID of the Employee relation and we renamed it **Manager\_ID**. In addition to adding the Simple attribute **Manager\_Experience** that was an attribute of the relation.

**2. SIGNS\_2 (Employment Contract Schema) :**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **ID** | **Employee\_ID** | **Start\_Date** | **End\_Date** | **Salary** | **Role** | **Date\_signed** |

Every **EMPLOYEE** has to sign an **EMPLOYMENT\_CONTRACT** and every **EMPLOYMENT\_CONTRACT** requires an **EMPLOYEE** to sign it. Hence the “SIGNS\_2” relationship links the **EMPLOYEE** Entity and **EMPLOYMENT\_CONTRACT** 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 chose the **EMPLOYMENT\_CONTRACT** relation in which we added, as a foreign key, the primary key ID of the Employee relation and we renamed it **Employee\_ID**.

**3**. **HAS ( Point\_Of\_Contact\_Schema):**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Email** | **Client\_ID** | **First\_Name** | **Middle\_Initial** | **Last\_Name** | **Phone\_Number** |

The mapping for this was already applied correctly, Where the Client\_ID is put as a foreign key in the Point\_Of\_Contact relation Schema

**4.** **HAS\_2 (Rent\_Contract Schema):**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **ID** | **Space\_ID** | **Start\_Date** | **End\_Date** | **Rent\_Cost** |

Every **RENT\_CONTRACT** requires a **SPACE**. The “Has\_2” relationship links **RENT\_CONTRACT** and **SPACE** entities. there is a Total participation from the **RENT\_CONTRACT** side. Hence, We chose the **RENT\_CONTRACT** schema in which we added, as a foreign key, the primary key ID of the **SPACE** relation and we renamed it **Space\_ID**.

**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:

* **WORKS\_AT**
* **CONTAINS\_1**
* **CONTAINS\_2**
* **PLAYS**
* **CONTAINS\_4**
* **RENTS**
* **SIGNS\_1.**

**1.** **WORKS\_AT (Employee Schema):**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| ID | Department\_ID | First\_Name | Middle\_Initial | Last\_Name | Date\_of\_Birth |
| Gender | Email | Phone\_Number | City | Street\_Name |  |

Many **EMPLOYEES** Work At one **DEPARTMENT**. The “Works\_At'' relationship links the **EMPLOYEE** entity and the **DEPARTMENT** entity. The **EMPLOYEE** entity is on the “many'' side. Thus, we add to its relation the foreign key ID which is the primary key of the **DEPARTMENT** entity and we rename it **Department\_ID**.

**2.** **CONTAINS\_1 (Department Schema):**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| ID | Floor\_ID | Manager\_ID | Manager \_experience | Name | NbEmployees |

Many **DEPARTMENTS** may be located on one **FLOOR**. Thus the “Contains\_1” relationship links the **FLOOR** entity and the **DEPARTMENT** entity. The **DEPARTMENT** entity is on the “many” side. Hence, we add to its relation the foreign key ID which is the primary key of the Floor entity and we rename it **Floor\_ID**.

**3.** **CONTAINS\_2 (Space Schema):**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **ID** | **Floor\_ID** | **Length** | **Width** | **Available** | **Type** |

A **FLOOR** may contain many **SPACES**. The “CONTAINS\_2” relationship links the **FLOOR** entity and the **SPACE** entity. The **SPACE**  entity is on the many sides of the relation. so, we add to its relation the foreign key ID which is the primary key of the **FLOOR** entity and we rename it **Floor\_ID**.

**4. PLAYS (Floor Schema):**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **ID** | **Music\_ID** | **Level** | **Total\_Spaces** | **Available\_Spaces** | **Total\_Departments** |

Some **FLOORS** may PLAY **MUSIC**. The “Contains\_4” relationship links the **FLOOR**  entity and the **MUSIC** entity. The **FLOOR** entity is on the “many” side of the relationship. So, we add to its relation the foreign key ID which is the primary key of the **MUSIC** entity and we rename it **Music\_ID**.

**6.** **SIGNS\_1 (Rent\_Contract Schema):**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **ID** | **Space\_ID** | **Email** | **Client\_ID** | **Start\_Date** | **End\_Date** | **Rent\_Cost** |

Some **POINT\_OF\_CONTACT**  may Sign one or more **RENT\_CONTRACT** . The “Signs\_1” relationship links the weak **Point\_Of\_Contact** entity and the **RENT\_CONTRACT** entity. The **RENT\_CONTRACT** entity is on the “many” side of the relation. so, we add to its relation the foreign key **Email** and **Client\_ID** which is the primary key of the **POINT\_OF\_CONTACT** entity.

## Step 5: Mapping of M:N Relationship Types

In this step, we are going to map the binary many-to-many relationships. For each many-to-many relationship we are going to create a new relation which includes, as foreign keys, the primary keys of all participating relations. Their combination will form the primary key of this newly created relation. We must also include any other simple attribute of the many-to-many relationship. The many-to-many relationships needed to be mapped are:

* **CONTAINS\_3**

**1. CONTAINS\_3**:

|  |  |
| --- | --- |
| **Floor\_ID** | **Event\_ID** |

Many **FLOORS** may contain many **EVENTS**. The “Contains\_3” relationship links the **FLOOR** entity and the **EVENT** entity. We create a new relation called “Contains\_3” that includes the primary keys of the **FLOOR** and **EVENT** entities. The primary key of the **FLOOR** entity, ID, is added to the “Contains\_3” relation and renamed **Floor\_ID**. Also the primary key of the **EVENT** entity, **ID**, is added and renamed **Event\_ID**. **The combination of both added keys represents the primary key of the “Contains\_3” relation and thus are underlined.**

## 

## Step 6: Mapping of Multivalued Attributes

In this step, we are going to map the multivalued attributes which we ignored before. For each multivalued attribute we create a new relation containing the related attribute and the primary key of the entity to which it belongs. Their combination will represent the primary key of the newly created relation. The only multi-valued attribute in our database belongs to the Department Entity and it is the **Department Service Offered**.

**1. Department\_Service\_Offered:**

|  |  |
| --- | --- |
| Department\_ID | Service\_Offered |

The multivalued attribute Service\_Offered belongs to the Department entity. To represent it, we create a relation called “**Department\_Service\_Offered**”. Its primary key is composed of **Department\_ID**, the primary key of the **Department** entity, and the attribute **Service\_Offered** which represents the multiple Services a **Department** can have.

**Step 7: Mapping of N-ary Relationship Types**

In this step, we are going to map the N-ary Relationship types. We should create a new relation containing the primary keys of all participating entities and any simple attributes of the relationship type. In our design **we do not have any N-ary relationship types**, so this step is not applicable here.

## 

## **Final Displays:**

1. **Employee:**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| ID | Department\_ID | First\_Name | Middle\_Initial | Last\_Name | Date\_of\_Birth |
| Gender | Email | Phone\_Number | City | Street\_Name |  |

1. **Employement\_Contract:**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| ID | Employee\_ID | Start\_Date | End\_Date | Salary | Role | Date\_signed |

1. **Department:**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| ID | Floor\_ID | Manager\_ID | Manager\_experience | Name | Nb\_Employees |

1. **Department\_Service\_Offered:**

|  |  |
| --- | --- |
| Department\_ID | Service\_Offered |

1. **Space:**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| ID | Floor\_ID | Length | Width | Available | Rent\_Cost | Type |

1. **Rent\_Contract:**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| ID | Space\_ID | Email | Client\_ID | Start\_Date | End\_Date | Rent\_Cost |

1. **Client:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ID | Business Name | Phone Number | Email Address | Rented\_Spaces\_Nb |
| City | Street\_Name |  | | |

1. **Point\_of\_Contact:**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Email | Client\_ID | First\_Name | Middle\_Initial | Last\_Name | Phone\_Number |

1. **Floor:**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| ID | Music\_ID | Level | Total\_Spaces | Available\_Spaces | Total\_Departments | Type |

1. **Parking\_Lot:**

|  |  |  |  |
| --- | --- | --- | --- |
| ID | Floor\_ID | Total\_Parking\_Spaces | Total\_Available\_Spaces |

1. **Music:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ID | Title | Composer | Genre | Duration |

1. **Event:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ID | Name | Sponsor | Start Date | End Date |

1. **Contains\_3:**

|  |  |
| --- | --- |
| **Floor\_ID** | **Event\_ID** |

# VI - Database Creation

## Step 1: Table Creation

After designing the ER diagram for Shopping Mall and mapping this diagram into relational database design, now it is time to start creating the actual tables for our database on the Oracle Database Server. We will start by creating all tables and then inserting data into these tables. Finally, we will execute some queries to display the importance of the database and especially in a Shopping Mall.

1. **Music**

**CREATE TABLE** MUSIC

(

ID **INT** NOT NULL,

TITLE **VARCHAR(30)** NOT NULL,

COMPOSER **VARCHAR(30)** NOT NULL,

GENRE **VARCHAR(30)** NOT NULL,

DURATION **VARCHAR(5)** ,

**PRIMARY KEY** (ID)

) ;

**Music Table Description**

|  |  |  |  |
| --- | --- | --- | --- |
| **Column Name** | **Data Type** | **Nullable** | **Primary Key** |
| **ID** | NUMBER | No | 1 |
| **TITLE** | VARCHAR2(30) | No |  |
| **COMPOSER** | VARCHAR2(30) | No |  |
| **GENRE** | VARCHAR2(30) | No |  |
| **DURATION** | VARCHAR2(5) | No |  |

1. **Floor**

**CREATE TABLE** FLOOR

(

ID **INT** NOT NULL,

MUSIC\_ID **INT** ,

FLOOR\_LEVEL **INT** NOT NULL,

TOTAL\_SPACES  **INT** NOT NULL  **CHECK** (TOTAL\_SPACES>=0),

AVAILABLE\_SPACES  **INT** NOT NULL **CHECK** (AVAILABLE\_SPACES>=0),

TOTAL\_DEPARTMENTS **INT** NOT NULL **CHECK** (TOTAL\_DEPARTMENTS>=0),

TYPE **VARCHAR**(15) NOT NULL,

**PRIMARY KEY**(ID),

**FOREIGN KEY** ( MUSIC\_ID ) **REFERENCES**  MUSIC( ID )

) ;

**Floor Table Description**

|  |  |  |  |
| --- | --- | --- | --- |
| **Column Name** | **Data Type** | **Nullable** | **Primary Key** |
| **ID** | NUMBER | No | 1 |
| **MUSIC\_ID** | NUMBER | Yes |  |
| **FLOOR\_LEVEL** | NUMBER | No |  |
| **TOTAL\_SPACES** | NUMBER | No |  |
| **AVAILABLE\_SPACES** | NUMBER | No |  |
| **TOTAL\_DEPARTMENTS** | NUMBER | No |  |
| **TYPE** | VARCHAR2 | No |  |

1. **Parking\_Lot**

**CREATE TABLE**  PARKING\_LOT

(

ID  **INT** NOT NULL,

FLOOR\_ID  **INT** NOT NULL,

TOTAL\_PARKING\_SPACES **INT** NOT NULL **CHECK**(TOTAL\_PARKING\_SPACES>=0),

TOTAL\_AVAILABLE\_SPACES  **INT** NOT NULL **CHECK** (TOTAL\_AVAILABLE\_SPACES>=0),

**PRIMARY KEY**(ID)

**FOREIGN KEY**(FLOOR\_ID) **REFERENCES** FLOOR(ID)

) ;

**Parking\_Lot Table Description**

|  |  |  |  |
| --- | --- | --- | --- |
| **Column Name** | **Data Type** | **Nullable** | **Primary key** |
| **ID** | NUMBER | No | 1 |
| **FLOOR\_ID** | NUMBER | No |  |
| **TOTAL\_PARKING\_SPACES** | NUMBER | No |  |
| **TOTAL\_AVAILABLE\_SPACES** | NUMBER | No |  |

1. **Department**

**CREATE TABLE** DEPARTMENT

(

ID **INT**  NOT NULL,

FLOOR\_ID **INT**  NOT NULL,

MANAGER\_ID **INT**  ,

MANAGER\_EXPERIENCE **INT**  ,

NAME **VARCHAR(30)** NOT NULL,

Nb\_EMPLOYEES **INT**  NOT NULL,

FOREIGN KEY (FLOOR\_ID) **REFERENCES**  FLOOR (ID),

PRIMARY KEY (ID)) ;

**Department Table Description**

|  |  |  |  |
| --- | --- | --- | --- |
| **Column Name** | **Data Type** | **Nullable** | **Primary Key** |
| **ID** | NUMBER | No | 1 |
| **FLOOR\_ID** | NUMBER | No |  |
| **MANAGER\_ID** | NUMBER | Yes |  |
| **MANAGER\_EXPERIENCE** | VARCHAR2(30) | Yes |  |
| **NAME** | NUMBER | No |  |
| **Nb\_EMPLOYEES** | NUMBER | No |  |

**Deartment**

1. **Department\_ Service\_Offered**

**CREATE TABLE** DEPARTMENT\_SERVICE\_OFFERED

(

DEPARTMENT\_ID **INT**  NOT NULL,

SERVICE\_OFFERED **VARCHAR(50)** NOT NULL,

FOREIGN KEY (DEPARTMENT\_ID) **REFERENCES** DEPARTMENT( ID ),

PRIMARY KEY (DEPARTMENT\_ID, SERVICE\_OFFERED)

);

**Department\_Service\_Offered Table Description**

|  |  |  |  |
| --- | --- | --- | --- |
| **Column Name** | **Data Type** | **Nullable** | **Primary Key** |
| **DEPARTMENT\_ID** | NUMBER | No | 1 |
| **SERVICE\_OFFERED** | VARCHAR2(50) | No | 2 |

1. **Employee:**

**CREATE TABLE** EMPLOYEE

(

ID **INT**  NOT NULL,

DEPARTMENT\_ID **INT**  ,

FIRST\_NAME **VARCHAR(15)**  NOT NULL,

MIDDLE\_INITIAL **VARCHAR(15)**  NOT NULL,

LAST\_NAME **VARCHAR(15)**  NOT NULL,

DATE\_OF\_BIRTH **DATE**  NOT NULL,

GENDER **CHAR(1)**  NOT NULL **CHECK** ( GENDER **IN** ('F', 'M')),

EMAIL **VARCHAR(50)**  ,

PHONE\_NUMBER **VARCHAR(10)** ,

CITY **VARCHAR(15)**  NOT NULL,

STREET\_NAME **VARCHAR(15)** NOT NULL,

**UNIQUE** (PHONE\_NUMBER),

**PRIMARY KEY** (ID),

**FOREIGN KEY**(DEPARTMENT\_ID) **REFERENCES** DEPARTMENT(ID)) ;

**Employee Table Description**

|  |  |  |  |
| --- | --- | --- | --- |
| **Column Name** | **Data Type** | **Nullable** | **Primary Key** |
| **ID** | NUMBER | No | **1** |
| **DEPARTMENT\_ID** | NUMBER | Yes |  |
| **FIRST\_NAME** | VARCHAR2(15) | No |  |
| **MIDDLE\_INITIAL** | CHAR (1) | No |  |
| **LAST\_NAME** | VARCHAR2(15) | No |  |
| **DATE\_OF\_BIRTH** | DATE | No |  |
| **GENDER** | CHAR (1) | No |  |
| **EMAIL** | VARCHAR2(50) | Yes |  |
| **PHONE\_NUMBER** | NUMBER | Yes |  |
| **CITY** | VARCHAR2(15) | No |  |
| **STREET\_NAME** | VARCHAR2(15) | No |  |

***After Creating the Employee Table, We will Alter the Department Table and Specify the Foreign key Constraint on Manager\_ID which we previously ignored since the Employee Table was not created then.***

**ALTER TABLE** DEPARTMENT **ADD** ( **FOREIGN KEY** ( MANAGER\_ID )

**REFERENCES** EMPLOYEE( ID ) ) ;

1. **Employment\_Contract**

**CREATE TABLE** EMPLOYMENT\_CONTRACT

(

ID **INT**  NOT NULL,

EMPLOYEE\_ID **INT**  NOT NULL,

START\_DATE **DATE**  NOT NULL,

END\_DATE **DATE**  NOT NULL,

SALARY **INT**  NOT NULL **CHECK** ( SALARY > 0 ),

ROLE **VARCHAR(50)** ,

DATE\_SIGNED **DATE** NOT NULL,

**FOREIGN KEY** (EMPLOYEE\_ID) **REFERENCES** EMPLOYEE (ID),

**PRIMARY KEY** (ID)

);

**Employement\_Contract Table Description**

|  |  |  |  |
| --- | --- | --- | --- |
| **Column Name** | **Data Type** | **Nullable** | **Primary Key** |
| **ID** | NUMBER | No | 1 |
| **EMPLOYEE\_ID** | NUMBER | No |  |
| **START\_DATE** | DATE | No |  |
| **END\_DATE** | DATE | No |  |
| **SALARY** | NUMBER | No |  |
| **ROLE** | VARCHAR(50) | Yes |  |
| **DATE\_SIGNED** | DATE | No |  |

1. **Client**

**CREATE TABLE** CLIENT

(

ID **INT** NOT NULL,

BUSINESS\_NAME **VARCHAR(30)**  NOT NULL,

PHONE\_NUMBER  **INT** ,

EMAIL\_ADDRESS **VARCHAR(50)** ,

RENTED\_SPACES\_NUMBER **INT** NOT NULL,

CITY **VARCHAR(30)**  NOT NULL,

STREET\_NAME **VARCHAR(30)** NOT NULL,

**PRIMARY KEY**(ID)

) ;

**Client Table Description**

|  |  |  |  |
| --- | --- | --- | --- |
| **Column Name** | **Data Type** | **Nullable** | **Primary Key** |
| **ID** | NUMBER | No | **1** |
| **BUSINESS\_NAME** | VARCHAR2(30) | No |  |
| **PHONE\_NUMBER** | NUMBER | Yes |  |
| **EMAIL\_ADDRESS** | VARCHAR2(50) | Yes |  |
| **RENTED\_SPACES\_**  **NUMBER** | NUMBER | No |  |
| **CITY** | VARCHAR2(30) | No |  |
| **STREET\_NAME** | VARCHAR2(30) | No |  |

1. **Point\_Of\_Contact**

**CREATE TABLE** POINT\_OF\_CONTACT

(

EMAIL **VARCHAR(50)**  NOT NULL,

CLIENT\_ID **INT** NOT NULL,

FIRST\_NAME **VARCHAR(15)** NOT NULL,

MIDDLE\_INITIAL **CHAR(1)** NOT NULL,

LAST\_NAME **VARCHAR(15)** NOT NULL,

PHONE\_NUMBER **INT** NOT NULL ,

**PRIMARY KEY**(EMAIL,CLIENT\_ID),

**FOREIGN KE**Y(CLIENT\_ID) **REFERENCES** CLIENT(ID)

);

**Point\_Of\_Contact Table Description**

|  |  |  |  |
| --- | --- | --- | --- |
| **Column Name** | **Data Type** | **Nullable** | **Primary Key** |
| **EMAIL** | VARCHAR2(50) | No | 1 |
| **CLIENT\_ID** | NUMBER | No | 2 |
| **FIRST\_NAME** | VARCHAR2(15) | No |  |
| **MIDDLE\_INITIAL** | CHAR(1) | No |  |
| **LAST\_NAME** | VARCHAR2(15) | No |  |
| **PHONE\_NUMBER** | NUMBER | No |  |

1. **Space**

**CREATE TABLE** SPACE

(

ID **INT** NOT NULL,

FLOOR\_ID **INT** NOT NULL,

LENGTH **FLOAT** NOT NULL **CHECK** (LENGTH>0),

WIDTH **FLOAT**  NOT NULL **CHECK** (WIDTH>0),

AVAILABLE **CHAR(1)** NOT NULL **CHECK**(AVAILABLE IN (‘Y’,’N’)),

RENT\_COST **INT** NOT NULL **CHECK** (RENT\_COST>0),

TYPE **VARCHAR(15)**,

**PRIMARY KEY**(ID),

**FOREIGN KEY** (FLOOR\_ID) **REFERENCES** FLOOR(ID)

);

**Space Description**

|  |  |  |  |
| --- | --- | --- | --- |
| **Column Name** | **Data Type** | **Nullable** | **Primary Key** |
| **ID** | NUMBER | No | 1 |
| **FLOOR\_ID** | NUMBER | No |  |
| **LENGTH** | NUMBER(6,2) | No |  |
| **WIDTH** | NUMBER(6,2) | No |  |
| **AVAILABLE** | CHAR(1) | No |  |
| **RENT\_COST** | NUMBER | No |  |
| **TYPE** | VARCHAR2(15) | Yes |  |

1. **Rent\_Contract:**

**CREATE TABLE** RENT\_CONTRACT

(

ID **INT**  NOT NULL,

SPACE\_ID **INT**  NOT NULL,

POINT\_OF\_CONTACT\_EMAIL **VARCHAR(30)** NOT NULL,

CLIENT\_ID **INT** NOT NULL,

START\_DATE **DATE**  NOT NULL,

END\_DATE **DATE**  NOT NULL,

RENT\_COST **INT**  NOT NULL,

**FOREIGN KEY** (SPACE\_ID) **REFERENCES** SPACE ( ID),

**FOREIGN KEY** (POINT\_OF\_CONTACT\_EMAIL, CLIENT\_ID) **REFERENCES** POINT\_OF\_CONTACT ( EMAIL, CLIENT\_ID ),

**PRIMARY KEY** ( ID )

) ;

**Rent\_Contract Table Description**

|  |  |  |  |
| --- | --- | --- | --- |
| **Column Name** | **Data Type** | **Nullable** | **Primary Key** |
| **ID** | NUMBER | No | 1 |
| **SPACE\_ID** | NUMBER | No |  |
| **POINT\_OF\_CONTACT\_EMAIL** | VARCHAR2(50) | No |  |
| **CLIENT\_ID** | NUMBER | No |  |
| **START\_DATE** | DATE | No |  |
| **END\_DATE** | DATE | No |  |
| **RENT\_COST** | NUMBER | No |  |

1. **Event**

**CREATE TABLE** EVENT

**(**

ID  **INT** NOT NULL**,**

NAME **VARCHAR(50)** NOT NULL**,**

SPONSOR  **VARCHAR(50) ,**

START\_DATE  **DATE** NOT NULL**,**

END\_DATE  **DATE** NOT NULL**,**

**PRIMARY KEY** (ID)

);

**Event Table Description**

|  |  |  |  |
| --- | --- | --- | --- |
| **Column Name** | **Data Type** | **Nullable** | **Primary Key** |
| **ID** | NUMBER | No | 1 |
| **Theme** | VARCHAR2(50) | No |  |
| **SPONSOR** | VARCHAR2(50) | Yes |  |
| **START\_DATE** | DATE | No |  |
| **END\_DATE** | DATE | No |  |

1. **Contains\_3**

**CREATE TABLE** CONTAINS\_3

(

FLOOR\_ID **INT** NOT NULL,

EVENT\_ID **INT** NOT NULL,

FOREIGN KEY ( FLOOR\_ID) REFERENCES FLOOR ( ID ),

**PRIMARY KEY** (FLOOR\_ID,EVENT\_ID)

);

**Contains\_3 Table Description**

|  |  |  |  |
| --- | --- | --- | --- |
| **Column Name** | **Data Type** | **Nullable** | **Primary Key** |
| **FLOOR\_ID** | NUMBER | No | 1 |
| **EVENT\_ID** | NUMBER | No | 2 |

## Step 2: Data Insertion

1. **Music**

---------------------------------------------------------------------------------------------------

**Insertion Format:**

**INSERT INTO** MUSIC **VALUES** ( ID , Title, Composer , Genre , Duration ) ;

---------------------------------------------------------------------------------------

**INSERT INTO** MUSIC **VALUES** (1, 'in my feelings', 'Drake', 'hip hop', '3:36' );

**INSERT INTO** MUSIC **VALUES** (2, 'Golden', 'Harry Styles', 'pop', '4:01' );

**INSERT INTO** MUSIC **VALUES** (3, 'are you with me', 'Lost Frequencies', 'electronic', '2:56' );

**INSERT INTO** MUSIC **VALUES** (4, 'a thousand times over', 'Ella Mai', 'R&B', '4:32' );

**INSERT INTO** MUSIC **VALUES** (5, 'Reality', 'Lost Frequencies', 'electronic', '3:09' );

**INSERT INTO** MUSIC **VALUES** (6, 'give a little', 'Maggie Rogers', 'Pop', '3:41' );

**INSERT INTO** MUSIC **VALUES** (7, 'In the end', 'Linkin Park', 'Rock', '3:04' );

**INSERT INTO** MUSIC **VALUES** (8, 'Fire on fire', 'Sam Smith', 'Pop', '5:49' );

**INSERT INTO** MUSIC **VALUES** (9, 'After hours', 'The Weekend', ' R&B ', '6:17' );

**INSERT INTO** MUSIC **VALUES** (10, 'Imagine', 'John Lennon', ' Rock ' , '4:02' );

**2) Floor**

* **We will First Insert the Floor data with 0 Total spaces, 0 available spaces and 0 Total departments**
* **Once we Insert the Spaces for each floor, we will update the total spaces number for each floor as well as the available spaces**
* **Once a client rents a space, we will update the number of available spaces**
* **Once we Insert the departments, we will update the number of total departments**

---------------------------------------------------------------------------------------------------

**Insertion Format:**

**INSERT INTO** FLOOR **VALUES** ( ID , Music\_ID, Floor\_Level , Total spaces , Available spaces , Total Departments, Type ) ;

---------------------------------------------------------------------------------------

**INSERT INTO** FLOOR **VALUES** ( 0 , 10 , -3 , 0 , 0 , 0 , ‘Parking Lot’ );

**INSERT INTO** FLOOR **VALUES** ( 1 , 10 , -2 , 0 , 0 , 0 , ‘Parking Lot’ );

**INSERT INTO** FLOOR **VALUES** ( 2 , 9 , -1 , 0 , 0 , 0 , ‘Parking Lot’ );

**INSERT INTO** FLOOR **VALUES** ( 3 , 8 , 0 , 0 , 0 , 0 , ‘Electronics’ ) ;

**INSERT INTO** FLOOR **VALUES** ( 4 , 7 , 1 , 0 , 0 , 0, ‘ Cloths ’ ) ;

**INSERT INTO** FLOOR **VALUES** ( 5 , 6 , 2 , 0 , 0 , 0 , ‘Restaurants’ ) ;

**INSERT INTO** FLOOR **VALUES** ( 6 , 5 , 3 , 0 , 0 , 0, ‘ Sports equipment ’ ) ;

**3) Parking\_Lot**

---------------------------------------------------------------------------------------------------

**Insertion Format:**

**INSERT INTO** PARKING\_LOT **VALUES** ( ID , Floor\_ID , Total spaces, Available Spaces );

---------------------------------------------------------------------------------------------------

**INSERT INTO** PARKING\_LOT **VALUES** ( 1 , 0 , 50 , 20 ) ;

**INSERT INTO** PARKING\_LOT **VALUES** ( 2 , 1 , 75 , 35 ) ;

**INSERT INTO** PARKING\_LOT **VALUES** ( 3 , 2 , 100 , 40 ) ;

**4) Space**

---------------------------------------------------------------------------------------------------

**Insertion Format:**

**INSERT INTO** SPACE **VALUES** ( ID , Floor\_ID , Length, Width, Available , Rent\_Cost, Type);

---------------------------------------------------------------------------------------------------

**INSERT INTO** SPACE **VALUES** ( 0 , 3 , 15 , 15 , ‘Y’ , 3375 , ‘Shop’ );

**INSERT INTO** SPACE **VALUES** ( 1 , 3 , 10 , 15 , ‘Y’ , 2250 , ‘Restaurant’ );

**INSERT INTO** SPACE **VALUES** ( 2 , 3 , 10 , 8 , ‘Y’ , 800 , ‘Shop’ );

**INSERT INTO** SPACE **VALUES** ( 3 , 3 , 20 , 25 , ‘Y’ , 5000 , ‘Shop’ );

**INSERT INTO** SPACE **VALUES** ( 4 , 3 , 15 , 12 , ‘Y’ , 1800 , ‘Shop’ );

**INSERT INTO** SPACE **VALUES** ( 5 , 3 , 3 , 2 , ‘Y’ , 600 , ‘Ad’ );

**INSERT INTO** SPACE **VALUES** ( 6 , 3 , 5 , 5 , ‘Y’ , 2500 , ‘Ad’ );

***After Inserting all the spaces for Floor 3, We update its Total\_Spaces as well as its available spaces.***

**UPDATE** FLOOR

**SET** Total\_Spaces = 7 , Available\_Spaces = 7

**WHERE** ID = 3;

**INSERT INTO** SPACE **VALUES** ( 7 , 4 , 10 , 8 , ‘Y’ , 800 , ‘Shop’ );

**INSERT INTO** SPACE **VALUES** ( 8 , 4 , 20 , 25 , ‘Y’ , 5000 , ‘Shop’ );

**INSERT INTO** SPACE **VALUES** ( 9 , 4 , 15 , 12 , ‘Y’ , 1800 , ‘Shop’ );

**INSERT INTO** SPACE **VALUES** ( 10 , 4 , 10 , 8 , ‘Y’ , 800 , ‘Shop’ );

**INSERT INTO** SPACE **VALUES** ( 11 , 4 , 20 , 25 , ‘Y’ , 5000 , ‘Shop’ );

**INSERT INTO** SPACE **VALUES** ( 12 , 4 , 15 , 12 , ‘Y’ , 1800 , ‘Shop’ );

**INSERT INTO** SPACE **VALUES** ( 13 , 4 , 15 , 12 , ‘Y’ , 1800 , ‘Shop’ );

**INSERT INTO** SPACE **VALUES** ( 14 , 4 , 3 , 2 , ‘Y’ , 600 , ‘Ad’ );

**INSERT INTO** SPACE **VALUES** ( 15 , 4 , 5 , 5 , ‘Y’ , 2500 , ‘Ad’ );

***After Inserting all the spaces for Floor 4, We update its Total\_Spaces as well as its available spaces.***

**UPDATE** FLOOR

**SET** Total\_Spaces = 9 , Available\_Spaces = 9

**WHERE** ID = 4;

**INSERT INTO** SPACE **VALUES** ( 16 , 5 , 10 , 8 , ‘Y’ , 800 , ‘Shop’ );

**INSERT INTO** SPACE **VALUES** ( 17 , 5 , 20 , 25 , ‘Y’ , 5000 , ‘Shop’ );

**INSERT INTO** SPACE **VALUES** ( 18 , 5 , 15 , 12 , ‘Y’ , 1800 , ‘Shop’ );

**INSERT INTO** SPACE **VALUES** ( 19 , 5 , 10 , 8 , ‘Y’ , 800 , ‘Shop’ );

**INSERT INTO** SPACE **VALUES** ( 20 , 5 , 20 , 25 , ‘Y’ , 5000 , ‘Shop’ );

**INSERT INTO** SPACE **VALUES** ( 21 , 5 , 15 , 12 , ‘Y’ , 1800 , ‘Shop’ );

**INSERT INTO** SPACE **VALUES** ( 22 , 5 , 3 , 2 , ‘Y’ , 600 , ‘Ad’ );

**INSERT INTO** SPACE **VALUES** ( 23 , 5 , 5 , 5 , ‘Y’ , 2500 , ‘Ad’ );

**INSERT INTO** SPACE **VALUES** ( 24 , 5 , 2 , 2 , ‘Y’ , 400 , ‘Ad’ );

***After Inserting all the spaces for Floor 5, We update its Total\_Spaces as well as its available spaces.***

**UPDATE** FLOOR

**SET** Total\_Spaces = 9 , Available\_Spaces = 9

**WHERE** ID = 5;

**INSERT INTO** SPACE **VALUES** ( 25 , 6 , 15 , 15 , ‘N’ , 3375 , ‘Restaurant’ );

**INSERT INTO** SPACE **VALUES** ( 26 , 6 , 10 , 15 , ‘N’ , 2250 , ‘Restaurant’ );

**INSERT INTO** SPACE **VALUES** ( 27 , 6 , 10 , 5 , ‘N’ , 1000 , ‘Restaurant’ );

**INSERT INTO** SPACE **VALUES** ( 28 , 6 , 20 , 20 , ‘N’ , 6000 , ‘Restaurant’ );

***After Inserting all the spaces for Floor 6, We update its Total\_Spaces as well as its available spaces.***

**UPDATE** FLOOR

**SET** Total\_Spaces = 4 , Available\_Spaces = 4

**WHERE** ID = 6;

**5) Department and Department\_Service\_Offered**

* **We will Insert the Departments**
* **Then Update the Floor with their Total Departments**
* **Manager\_ID and His/her years of experience is Inserted as NULL, we will update them once we Insert the Employee Entity and Assign each department a Manager**
* **We will also Insert the Services offered by the Department**

---------------------------------------------------------------------------------------------------

**Insertion Format:**

**INSERT INTO** DEPARTMENT **VALUES** ( ID , Floor\_ID , Manager\_ID, years of experience, Name , number of employees );

---------------------------------------------------------------------------------------

**Insertion Format:**

**INSERT INTO** DEPARTMENT\_SERVICE\_OFFERED **VALUES** ( Department\_ID , Service\_Offered);

---------------------------------------------------------------------------------------

**INSERT INTO** DEPARTMENT **VALUES** ( 0 , 3 , NULL, NULL, “Customer Service” , 0 );

**INSERT INTO** DEPARTMENT\_SERVICE\_OFFERED **VALUES** ( 0 , ‘Customer Feedback Collection and Analysis’ );

**INSERT INTO** DEPARTMENT\_SERVICE\_OFFERED **VALUES** ( 0 , ‘Answering client or customer questions’ );

---------------------------------------------------------------------------------------

**INSERT INTO** DEPARTMENT **VALUES** ( 1 , 3 , NULL, NULL, “Security” , 0 );

**INSERT INTO** DEPARTMENT\_SERVICE\_OFFERED **VALUES** ( 1 , ‘monitoring surveillance equipment’);

**INSERT INTO** DEPARTMENT\_SERVICE\_OFFERED **VALUES** ( 1 , ‘inspecting buildings, equipment, and access points’ );

**INSERT INTO** DEPARTMENT\_SERVICE\_OFFERED **VALUES** ( 1 , ‘Crime prevention and detection’ );

---------------------------------------------------------------------------------------

***After Inserting all the Departments for Floor 3, We update the Floor Total departments Attribute.***

**UPDATE** FLOOR

**SET** Total\_Departments = 2

**WHERE** ID = 3;

---------------------------------------------------------------------------------------

**INSERT INTO** DEPARTMENT **VALUES** ( 2 , 4 , NULL, NULL, “Accounting & Finance” , 0 );

**INSERT INTO** DEPARTMENT\_SERVICE\_OFFERED **VALUES** ( 2 , ‘Billing clients’ );

**INSERT INTO** DEPARTMENT\_SERVICE\_OFFERED **VALUES** ( 2 , ‘Managing payroll’);

**INSERT INTO** DEPARTMENT\_SERVICE\_OFFERED **VALUES** ( 2 , ‘Tracking assets and expenditures’ );

**INSERT INTO** DEPARTMENT\_SERVICE\_OFFERED **VALUES** ( 2 , ‘Keeping track of critical tax documents’ );

---------------------------------------------------------------------------------------

**INSERT INTO** DEPARTMENT **VALUES** ( 3 , 4 , NULL, NULL, “Promotions & Marketing” , 0 );

**INSERT INTO** DEPARTMENT\_SERVICE\_OFFERED **VALUES** ( 3 , ‘Managing the Advertisements and public relations’ );

**INSERT INTO** DEPARTMENT\_SERVICE\_OFFERED **VALUES** ( 3 , ‘Set the Spaces prices’ );

**INSERT INTO** DEPARTMENT\_SERVICE\_OFFERED **VALUES** ( 3 , ‘Sponsor Events’ );

---------------------------------------------------------------------------------------

***After Inserting all the Departments for Floor 4, We update the Floor Total departments Attribute.***

**UPDATE** FLOOR

**SET** Total\_Departments = 2

**WHERE** ID = 4;

---------------------------------------------------------------------------------------

**INSERT INTO** DEPARTMENT **VALUES** ( 4 , 6 , NULL, NULL, “Janitorial Services” , 0 );

**INSERT INTO** DEPARTMENT\_SERVICE\_OFFERED **VALUES** ( 4 , ‘Clean building floors’ );

**INSERT INTO** DEPARTMENT\_SERVICE\_OFFERED **VALUES** ( 4 , ‘Operate on sections under maintenance’ );

---------------------------------------------------------------------------------------

***After Inserting all the Departments for Floor 6, We update the Floor Total departments Attribute.***

**UPDATE** FLOOR

**SET** Total\_Departments = 1

**WHERE** ID = 6;

**6) Employee and Employment Contract**

* **For every employee, we will immediately insert his contract**
* **Later after inserting all the employees, we will choose a manager for each department and assign every employee his department**

---------------------------------------------------------------------------------------------------

**Insertion Format:**

**INSERT INTO** EMPLOYEE **VALUES** ( ID , Department\_ID, First\_Name, Middle\_Initial, Last\_Name, Date of birth, Gender, email, phone number, City, Street\_name );

---------------------------------------------------------------------------------------

**Insertion Format:**

**INSERT INTO** EMPLOYEMENT\_CONTRACT **VALUES** ( ID , Employee\_ID, Start\_Date, End\_Date, Salary, Role, Date\_Signed );

---------------------------------------------------------------------------------------

**---------------------------------------------------------------------------------------------------------------------**

**INSERT INTO** EMPLOYEE **VALUES** ( 1000 , 0 , ‘Charbel’ , ‘P’ , ‘Younis’ , ‘8/22/2000’ , ‘M’ , ‘charbel.younis@lau.edu’ , ‘71448040’ , ‘Beirut’ , ‘Badaro’ );

**INSERT INTO** EMPLOYMENT\_CONTRACT **VALUES** ( 1 , 1000 , ‘1-1-2020’ , ‘1-1-2021’ , 7000, ‘Customer Service Manager’ , ’11-16-2020’ );

**---------------------------------------------------------------------------------------------------------------------**

**INSERT INTO** EMPLOYEE **VALUES** ( 1001 , 0 , ‘Georges’ , ‘F’ , ‘Freiha’ , ‘6-14-2000’ , ‘M’ , ‘georges.freiha@lau.edu’ , ‘03351717’ , ‘Matn’ , ‘Jdeideh’ );

**INSERT INTO** EMPLOYMENT\_CONTRACT **VALUES** ( 2 , 1001 , ‘2-7-2020’ , ‘2-7-2021’ , 7000, ‘Customer Service Engineer’ , ’11-16-2020’);

**---------------------------------------------------------------------------------------------------------------------**

**INSERT INTO** EMPLOYEE **VALUES** ( 1002 , 0 , ‘Bahaa’ , ‘Y’ , ‘Thebian’ , ‘10/9/2001’ , ‘M’ , ‘bahaa.thebian@lau.edu’ , ‘79313543’ , ‘Baakline’ , ‘Al-Wata’ );

**INSERT INTO** EMPLOYMENT\_CONTRACT **VALUES** ( 3 , 1002 , ‘3-6-2020’ , ‘3-6-2022’ , 7000,‘Customer Service Representative’ , ’11-16-2020’ );

**---------------------------------------------------------------------------------------------------------------------**

**INSERT INTO** EMPLOYEE **VALUES** ( 1003 , 1 , ‘Chris’ , ‘M’ , ‘Chalhoub’ , ‘2/2/2000’ , ‘M’ , ‘chris.chalhoub@lau.edu’ , ‘70000180’ , ‘Matn’ , ‘Mansourieh’ );

**INSERT INTO** EMPLOYMENT\_CONTRACT **VALUES** ( 4 , 1003 , ‘11-24-2020’ ,‘11-24-2021’ , 7000, ‘Security Manager’ , ’11-16-2020’ );

**---------------------------------------------------------------------------------------------------------------------**

**INSERT INTO** EMPLOYEE **VALUES** ( 1004 , 1 , ‘Jones’ , ‘J’ , ‘Johnson’ , ‘2/9/1997’ , ‘M’ , ‘jones.johnson@shoppingmall.com’ , ‘03123456’ , ‘Beirut’ , ‘Bliss’ );

**INSERT INTO** EMPLOYMENT\_CONTRACT **VALUES** ( 5 , 1004 , ‘5-19-2020’ , ‘5-19-2023’ , 5000,‘Security Agent’ , ’11-17-2020’ );

**---------------------------------------------------------------------------------------------------------------------**

**INSERT INTO** EMPLOYEE **VALUES** ( 1005 , 1 , ‘Johnys’ , ‘K’ , ‘Freddy’ , ‘8/2/2001’ , ‘M’ , ‘johnys.freddy@shoppingmall.com’ , ‘70123321’ , ‘Beirut’ , ‘Downtown’ );

**INSERT INTO** EMPLOYMENT\_CONTRACT **VALUES** ( 6 , 1005 , ‘7-18-2020’ , ‘7-18-2021’ , 4000, ‘Safety Advisor’ , ’11-17-2020’ );

**---------------------------------------------------------------------------------------------------------------------**

**INSERT INTO** EMPLOYEE **VALUES** ( 1006 , 2 , ‘Sukwando’ , ‘F’ , ‘Smith’ , ‘10/4/1997’ , ‘M’ , ‘sukwando.smith@shoppingmall.com’ , ‘80909090’ , ‘Matn’ , ‘Bourj-Hammoud’ );

**INSERT INTO** EMPLOYMENT\_CONTRACT **VALUES** ( 7 , 1006 , ‘3-19-2020’ , ‘3-19-2022’ , 4000, ‘Accounting Manager’ , ’11-17-2020’ );

**---------------------------------------------------------------------------------------------------------------------**

**INSERT INTO** EMPLOYEE **VALUES** ( 1007 , 2 , ‘Lea’ , ‘O’ , ‘Abrahimis’ , ‘10/10/1999’ , ‘F’ , ‘lea.abrahimis@shoppingmall.com’ , ‘71566968’ , ‘Matn’ , ‘Bourj-Hammoud’ );

**INSERT INTO** EMPLOYMENT\_CONTRACT **VALUES** ( 8 , 1007 , ‘5-1-2020’ , ‘5-1-2021’ , 4000, ‘Financial Controller’ , ’11-18-2020’ );

**---------------------------------------------------------------------------------------------------------------------**

**INSERT INTO** EMPLOYEE **VALUES** ( 1008 , 2 , ‘Roberta’ , ‘L’ , ‘James’ , ‘10/1/2000’ , ‘F’ , ‘roberta.james@shoppingmall.com’ , ‘78963014’ , ‘Beirut’ , ‘Barbir’ );

**INSERT INTO** EMPLOYMENT\_CONTRACT **VALUES** ( 9 , 1008 , ‘9-25-2020’ , ‘9-25-2021’ , 4000, ‘Accountant’ , ’11-18-2020’ );

**---------------------------------------------------------------------------------------------------------------------**

**INSERT INTO** EMPLOYEE **VALUES** ( 1009 , 3 , ‘Suksuk’ , ‘M’ , ‘Jordan’ , ‘1/18/1999’ , ‘M’ , ‘suksuk.jordan@shoppingmall.com’ , ‘01377777’ , ‘Beirut’ , ‘Hamra’ );

**INSERT INTO** EMPLOYMENT\_CONTRACT **VALUES** ( 10 , 1009 , ‘7-13-2020’ , ‘7-13-2021’ , 6000 , ‘Marketing Manager’ , ’11-18-2020’);

**---------------------------------------------------------------------------------------------------------------------**

**INSERT INTO** EMPLOYEE **VALUES** ( 1010 , 3 , ‘Imad’ , ‘Y’ , ‘Mansour’ , ‘8/7/1999’ , ‘M’ , ‘imad.mansour@shoppingmall.com’ , ‘71655990’ , ‘Beirut’ , ‘Saray’ );

**INSERT INTO** EMPLOYMENT\_CONTRACT **VALUES** ( 11 , 1010 , ‘5-11-2020’ , ‘5-11-2021’ , 5500, ‘Marketing Consultant’ , ’11-19-2020’ );

**---------------------------------------------------------------------------------------------------------------------**

**INSERT INTO** EMPLOYEE **VALUES** ( 1011 , 4 , ‘Jason’ , ‘T’ , ‘Paul’ , ‘1/10/2000’ , ‘M’ , ‘jason.paul@shoppingmall.com’ , ‘03444534’ , ‘Beirut’ , ‘Airport’ );

**INSERT INTO** EMPLOYMENT\_CONTRACT **VALUES** ( 12 , 1011 , ‘8-12-2020’ , ‘8-12-2021’ , 3000, ‘Janitorial Manager’ , ’11-19-2020’ );

**---------------------------------------------------------------------------------------------------------------------**

**INSERT INTO** EMPLOYEE **VALUES (** 1012**,** 4 **,** ‘Moussa’ **,** ‘S’ **, ‘**Haidous**’ , ‘**9-10-2001**’ ,** ‘M’ **, ‘**moussa.haidous@lau.edu**’ , ‘**76696952’ **,** ‘Beirut’ **,** ‘Hamra’ );

**INSERT INTO** EMPLOYMENT\_CONTRACT **VALUES** ( 13 , 1012 , ‘9-11-2020’ , ‘9-11-2021’ , 4000, ‘ Janitorial advisor’ , ’11-19-2020’ );

**---------------------------------------------------------------------------------------------------------------------**

***After Inserting all the Employees and binding them to a contract, We promote some of them to be managers of our departments.***

**UPDATE** DEPARTMENT

**SET** Manager\_ID = 1000 , Manager\_experience = 3, Nb\_Employees = 3

**WHERE** ID = 0;

**UPDATE** DEPARTMENT

**SET** Manager\_ID = 1001 , Manager\_experience = 4 , Nb\_Employees = 3

**WHERE** ID = 1;

**UPDATE** DEPARTMENT

**SET** Manager\_ID = 1002 , Manager\_experience = 5 , Nb\_Employees = 3

**WHERE** ID = 2;

**UPDATE** DEPARTMENT

**SET** Manager\_ID = 1003 , Manager\_experience = 3 , Nb\_Employees = 2

**WHERE** ID = 3;

**UPDATE** DEPARTMENT

**SET** Manager\_ID = 1012 , Manager\_experience = 5 , Nb\_Employees = 2

**WHERE** ID = 4;

**7) Client and Point of Contact**

* **For every client inserted, we will follow it with their point of contact.**

---------------------------------------------------------------------------------------------------

**Insertion Format:**

**INSERT INTO** CLIENT **VALUES** ( ID , Business\_Name, Phone\_Number, Email, Number of rented spaces, City , Street\_Name);

---------------------------------------------------------------------------------------

**Insertion Format:**

**INSERT INTO** POINT\_OF\_CONTACT **VALUES** ( Email, Client\_ID , First\_Name, Middle\_Initial, Last\_Name, Phone number );

**---------------------------------------------------------------------------------------------------------------------**

**INSERT INTO** CLIENT **VALUES (** 1 , ‘ZARA’ , ‘01293949’ , ‘zaralebanon@gmail.com’ , 0 , ‘Beirut’ , ‘Kaskas Street’ );

**INSERT INTO** POINT\_OF\_CONTACT **VALUES ( ‘**joeyounis@gmail.com’ , 1 , ‘Joe’ , ‘C’ , ‘Younis’ , ‘03493929’ );

**---------------------------------------------------------------------------------------------------------------------INSERT INTO** CLIENT **VALUES (** 2 , ‘MANGO’ , ‘01737482’ , ‘mangolebanon@gmail.com’ , 0 , ‘Beirut’ , ‘Hamra Street’ );

**INSERT INTO** POINT\_OF\_CONTACT **VALUES (** ‘charbelsuksuk@gmail.com’ , 2 , ‘Charbel’ , ‘F’ , ‘Suksuk’ , ‘71938483’ );

**---------------------------------------------------------------------------------------------------------------------**

**INSERT INTO** CLIENT **VALUES (** 3 , ‘GS’ , ‘01555234’ , ‘gslebanon@outlook.com’ , 0 , ‘Beirut’ , ‘Makdessi Street’ );

**INSERT INTO** POINT\_OF\_CONTACT **VALUES (** ‘iambahas@gmail.com’ , 3 , ‘Bahas’ , ‘H’ , ‘Sukwando’ , ‘81229339’ );

**---------------------------------------------------------------------------------------------------------------------**

**INSERT INTO** CLIENT **VALUES (** 4 , ‘AMERICAN EAGLE OUTFITTERS’ , ‘01290949’ , ‘aelebanon@gmail.com’ , 0 , ‘Beirut’ , ‘Barbir’ );

**INSERT INTO** POINT\_OF\_CONTACT **VALUES (** ‘dylanjones@gmail.com’ , 4 , ‘Dylan’ , ‘J’ , ‘Jones’ , ‘03293920’ );

**---------------------------------------------------------------------------------------------------------------------**

**INSERT INTO** CLIENT **VALUES (** 5 , ‘PINKBERRY’ , ‘04123321’ , ‘pinkberrylebanon@gmail.com’ , 0 , ‘Beirut’ , ‘Verdun Street’ );

**INSERT INTO** POINT\_OF\_CONTACT **VALUES** ( ‘stonecold@gmail.com’ , 5 , ‘Steve’ , ‘C’ , ‘Austin’ , ‘71442992’ );

**---------------------------------------------------------------------------------------------------------------------**

**INSERT INTO** CLIENT **VALUES** ( 6 , ‘ATAMIAN WATCHES’ , ‘04555555’ , ‘atamianwatches@gmail.com’ , 0 , ‘El Metn’ , ‘Zalka Street’ );

**INSERT INTO** POINT\_OF\_CONTACT **VALUES** ( ‘andrewabb@gmail.com’ , 6 , ‘Andrew’ , ‘A’ , ‘Abboud’ , ‘81332950’ );

**---------------------------------------------------------------------------------------------------------------------**

**INSERT INTO** CLIENT **VALUES (** 7 , ‘FITNESS ZONE’ , ‘01293649’ , ‘mangolebanon@gmail.com’ , 0 , ‘Beirut’ , ‘Hamra Street’ );

**INSERT INTO** POINT\_OF\_CONTACT **VALUES (** ‘alexandrekr@gmail.com’ , 7 , ‘Alexandre’ , ‘G’ , ‘Krikorian’ , ‘70993848’ );

**---------------------------------------------------------------------------------------------------------------------**

**INSERT INTO** CLIENT **VALUES (** 8 , ‘ZARA’ , ‘01293949’ , ‘zaralebanon@gmail.com’ , 0 , ‘Beirut’ , ‘Kaskas Street’ );

**INSERT INTO** POINT\_OF\_CONTACT **VALUES (** ‘charbelhad@gmail.com’ , 8 , ‘Charbel’ , ‘P’ , ‘Haddad’ , ‘03443921’ );

**---------------------------------------------------------------------------------------------------------------------**

**INSERT INTO** CLIENT **VALUES (** 9 , ‘AIZONE’ , ‘01123456’ , ‘aizoneleb@hotmail.com’ , 0 , ‘Beirut’ , ‘Jones Street’ ) ;

**INSERT INTO** POINT\_OF\_CONTACT **VALUES (** ‘joerizk@gmail.com’ , 9 , ‘Joe’ , ‘M’ , ‘Rizk’ , ‘03473923’ );

**---------------------------------------------------------------------------------------------------------------------**

**INSERT INTO** CLIENT **VALUES** ( 10 , ‘ROMANCE TIMES’ , ‘01898989’ , ‘romancetimesleb@gmail.com’ , 0 , ‘Beirut’ , ‘Al Sanayeh Street’ );

**INSERT INTO** POINT\_OF\_CONTACT **VALUES (** ‘charbelrah@gmail.com’ , 10 , ‘Charbel’ , ‘C’ , ‘Rahal’ , ‘03455529’ );

**---------------------------------------------------------------------------------------------------------------------**

**INSERT INTO** CLIENT **VALUES (** 11 , ‘MALAK EL TAWOUK’ , ‘01292929’ , ‘malakeltawouklebanon@gmail.com’ , 0 , ‘El Matn’ , ‘Sagesse Street’ );

**INSERT INTO** POINT\_OF\_CONTACT **VALUES (** ‘touficsaade@gmail.com’ , 11 , ‘Toufic’ , ‘K’ , ‘Saade’ , ‘03491629’ );

**---------------------------------------------------------------------------------------------------------------------**

**INSERT INTO** CLIENT **VALUES (** 12 , ‘BURGER KING’ , ‘01333294’ , ‘bklebanon@gmail.com’ , 0 , ‘Beirut’ , ‘Kraytem’ );

**INSERT INTO** POINT\_OF\_CONTACT **VALUES (** ‘vanessa@gmail.com’ , 12 , ‘Vanessa’ , ‘N’ , ‘Nunez’ , ‘71738489’ );

**---------------------------------------------------------------------------------------------------------------------**

**INSERT INTO** CLIENT **VALUES (** 13 , ‘ROADSTER DINER’ , ‘01297777’ , ‘rdlebanon@hotmail.com’ , 0 , ‘Beirut’ , ‘Hamra Street’ );

**INSERT INTO** POINT\_OF\_CONTACT **VALUES (** ‘chrisbarakat@gmail.com’ , 13 , ‘Christopher’ , ‘G’ , ‘Barakat’ , ‘71922999’ );

**---------------------------------------------------------------------------------------------------------------------**

**8) Rent\_Contract:**

* **Here we will Insert all the Rent Contracts,**
* **Then update if the space is still available or not,**
* **as well as the available spaces for the Floor attribute**
* **and the Rented\_Spaces\_Number for the Client**

**---------------------------------------------------------------------------------------------------------------------**

**Insertion Format:**

**INSERT INTO** RENT\_CONTRACT **VALUES** ( ID , Space\_ID , Point\_Of\_Contact\_Email , Client\_ID, Start\_Date , End\_Date, Agreed\_Rent\_Cost);

**---------------------------------------------------------------------------------------------------------------------**

**INSERT INTO** RENT\_CONTRACT **VALUES** ( 1 , 2 , ‘joeyounis@gmail.com’ , 1 , ‘09-12-2021’ , ’09-12-2023’ , 750 );

**UPDATE** SPACE

**SET** Available = ‘N’

**WHERE** ID = 2 ;

**UPDATE** CLIENT

**SET** Rented\_Spaces\_Number = 1

**WHERE** ID = 1 ;

**--------------------------------------------------------------------------------------------------------------------**

**INSERT INTO** RENT\_CONTRACT **VALUES** ( 2 , 4 , ‘charbelsuksuk@gmail.com’ , 2 , ‘10-12-2020’ , ’10-12-2022’ , 1000 );

**UPDATE** SPACE

**SET** Available = ‘N’

**WHERE** ID = 4 ;

**UPDATE** CLIENT

**SET** Rented\_Spaces\_Number = 1

**WHERE** ID = 2 ;

**---------------------------------------------------------------------------------------------------------------------**

**INSERT INTO** RENT\_CONTRACT **VALUES** ( 3 , 6 , ‘iambahas@gmail.com’ , 3 , ‘05-12-2021’ , ’05-12-2024’ , 2000 );

**UPDATE** SPACE

**SET** Available = ‘N’

**WHERE** ID = 6 ;

**UPDATE** CLIENT

**SET** Rented\_Spaces\_Number = 1

**WHERE** ID = 3 ;

**---------------------------------------------------------------------------------------------------------------------**

**INSERT INTO** RENT\_CONTRACT **VALUES** ( 4 , 3 , ‘dylanjones@gmail.com’ , 4 , ‘09-12-2021’ , ’09-12-2023’ , 4500);

**UPDATE** SPACE

**SET** Available = ‘N’

**WHERE** ID = 3 ;

**UPDATE** CLIENT

**SET** Rented\_Spaces\_Number = 1

**WHERE** ID = 4 ;

**---------------------------------------------------------------------------------------------------------------------**

**---------------------------------------------------------------------------------------------------------------------**

***After all Rent\_Contracts for Floor 3 has been inserted, we update the total available spaces.***

**UPDATE** Floor

**SET** AVAILABLE\_SPACES = 3

**WHERE** ID = 3 ;

**---------------------------------------------------------------------------------------------------------------------**

**---------------------------------------------------------------------------------------------------------------------**

**INSERT INTO** RENT\_CONTRACT **VALUES** ( 5 , 12 , ‘stonecold@gmail.com’ , 5 , ‘04-09-2021’ , ’04-12-2023’ , 1750 );

**UPDATE** SPACE

**SET** Available = ‘N’

**WHERE** ID = 12 ;

**UPDATE** CLIENT

**SET** Rented\_Spaces\_Number = 1

**WHERE** ID = 5 ;

**---------------------------------------------------------------------------------------------------------------------**

**INSERT INTO** RENT\_CONTRACT **VALUES** ( 6 , 9 , ‘andrewabb@gmail.com’ , 6 , ‘02-01-2021’ , ’09-12-2023’ , 1250 );

**UPDATE** SPACE

**SET** Available = ‘N’

**WHERE** ID = 9 ;

**UPDATE** CLIENT

**SET** Rented\_Spaces\_Number = 1

**WHERE** ID = 6 ;

**---------------------------------------------------------------------------------------------------------------------**

**INSERT INTO** RENT\_CONTRACT **VALUES** ( 7 , 13 ‘alexandrekr@gmail.com’ , 7 , ‘05-12-2021’ , ’09-12-2024’ , 1500 );

**UPDATE** SPACE

**SET** Available = ‘N’

**WHERE** ID = 13 ;

**UPDATE** CLIENT

**SET** Rented\_Spaces\_Number = 1

**WHERE** ID = 7 ;

**---------------------------------------------------------------------------------------------------------------------**

**---------------------------------------------------------------------------------------------------------------------**

***After all Rent\_Contracts for Floor 4 has been inserted, we update the total available spaces***

**UPDATE** Floor

**SET** AVAILABLE\_SPACES = 6 ;

**WHERE** ID = 4 ;

**---------------------------------------------------------------------------------------------------------------------**

**---------------------------------------------------------------------------------------------------------------------**

**INSERT INTO** RENT\_CONTRACT **VALUES** ( 8 , 18 ‘charbelhad@gmail.com’ , 8 , ‘01-10-2021’ , ’09-12-2026’ , 1500 );

**UPDATE** SPACE

**SET** Available = ‘N’

**WHERE** ID = 18 ;

**UPDATE** CLIENT

**SET** Rented\_Spaces\_Number = 1

**WHERE** ID = 8 ;

**---------------------------------------------------------------------------------------------------------------------**

**INSERT INTO** RENT\_CONTRACT **VALUES** ( 9 , 17 , ‘joerizk@gmail.com’ , 9 , ‘08-01-2021’ , ’08-01-2025’ , 4250 );

**UPDATE** SPACE

**SET** Available = ‘N’

**WHERE** ID = 17 ;

**UPDATE** CLIENT

**SET** Rented\_Spaces\_Number = 1

**WHERE** ID = 9 ;

**---------------------------------------------------------------------------------------------------------------------**

**INSERT INTO** RENT\_CONTRACT **VALUES** ( 10 , 20 , ‘charbelrah@gmail.com’ , 10 , ‘09-12-2021’, ’09-12-2023’ , 5000 );

**UPDATE** SPACE

**SET** Available = ‘ N ’

**WHERE** ID = 20 ;

**UPDATE** CLIENT

**SET** Rented\_Spaces\_Number = 1

**WHERE** ID = 10;

**---------------------------------------------------------------------------------------------------------------------**

**---------------------------------------------------------------------------------------------------------------------**

***After all Rent\_Contracts for Floor 5 has been inserted, we update the total available spaces.***

**UPDATE** Floor

**SET** AVAILABLE\_SPACES = 6

**WHERE** ID = 5 ;

**---------------------------------------------------------------------------------------------------------------------**

**---------------------------------------------------------------------------------------------------------------------**

**INSERT INTO** RENT\_CONTRACT **VALUES** ( 11 , 26 , ‘touficsaade@gmail.com’ , 11 , ’12-12-2018’ , ’03-12-2020’ , 2250 );

**UPDATE** SPACE

**SET** Available = ‘N’

**WHERE** ID = 26 ;

**UPDATE** CLIENT

**SET** Rented\_Spaces\_Number = 1

**WHERE** ID = 11;

**---------------------------------------------------------------------------------------------------------------------**

**INSERT INTO** RENT\_CONTRACT **VALUES** ( 12 , 25 , ‘iambahas@gmail.com’ , 3 , ’12-12-2018’ , ’12-27-2020’ , 2250 );

**UPDATE** SPACE

**SET** Available = ‘N’

**WHERE** ID = 25 ;

**UPDATE** CLIENT

**SET** Rented\_Spaces\_Number = 2

**WHERE** ID = 3;

**---------------------------------------------------------------------------------------------------------------------**

**---------------------------------------------------------------------------------------------------------------------**

***After all Rent\_Contracts for Floor 6 has been inserted, we update the total available spaces.***

**UPDATE** Floor

**SET** AVAILABLE\_SPACES = 1

**WHERE** ID = 6 ;

**---------------------------------------------------------------------------------------------------------------------**

**INSERT INTO** RENT\_CONTRACT **VALUES** ( 13 , 1 , ‘chrisbarakat@gmail.com’ , 13 , ’12-12-2019’ , ’01-01-2020’ , 2250 );

**UPDATE** SPACE

**SET** Available = ‘N’

**WHERE** ID = 1 ;

**UPDATE** CLIENT

**SET** Rented\_Spaces\_Number = 1

**WHERE** ID = 13;

**9) Event**

**---------------------------------------------------------------------------------------------------------------------**

**Insertion Format:**

**INSERT INTO** EVENT **VALUES** ( ID , Theme , Sponsor, Start\_Date, End\_Date);

**---------------------------------------------------------------------------------------------------------------------**

**INSERT INTO** EVENT **VALUES(**101 , 'Christmas decorations' , 'Mall shareholders' , '11-28-2020', '01-08-2021');

**INSERT INTO** EVENT **VALUES (**102 , 'Halloween', 'Mall shareholders' , '10-25-2020' , '11-01-2020');

**INSERT INTO** EVENT **VALUES (**103 ,'Car Giveaway', 'Toyota car company' , '4-4-2021' , '4-10-2021');

**INSERT INTO** EVENT **VALUES (**104 , 'Red cross donation', 'Red Cross' , '11-28-2020', '01-08-2021');

**INSERT INTO** EVENT **VALUES (**105 , 'Valentine', 'Mall Shareholders' , '02-10-2021' , '02-18-2021');

**INSERT INTO** EVENT **VALUES (**106 , 'Easter', 'Mall Shareholders' , '04-12-2021' , '04-15-2021');

**INSERT INTO** EVENT **VALUES (**107 ,'Black Friday', 'Mall Shareholders' , '11-23-2021' , '11-28-2021');

**INSERT INTO** EVENT **VALUES (**108 , 'Fashion week', 'Clothes Shops' , '08-15-2021' , '08-23-2021');

**INSERT INTO** EVENT **VALUES (**109 , 'Mothers Day', 'jewelry Shops' , '03-18-2021' , '03-23-2021');

**INSERT INTO** EVENT **VALUES (**110 , 'Perfume Testing', 'Lacoste Perfume Shop' , '03-02-2021','03-03-2021');

**10) Contains\_3**

**---------------------------------------------------------------------------------------------------------------------**

**Insertion Format:**

**INSERT INTO** CONTAINS\_3 **VALUES** ( Floor\_ID, Event\_ID);

**---------------------------------------------------------------------------------------------------------------------**

**INSERT INTO** CONTAINS\_3 **VALUES (**3 , 101);

**INSERT INTO** CONTAINS\_3 **VALUES (**4 , 101);

**INSERT INTO** CONTAINS\_3 **VALUES (**5 , 101);

**INSERT INTO** CONTAINS\_3 **VALUES (**6 , 101);

**INSERT INTO** CONTAINS\_3 **VALUES (**3 , 104);

**INSERT INTO** CONTAINS\_3 **VALUES (**4 , 104);

**INSERT INTO** CONTAINS\_3 **VALUES (**5 , 104);

**INSERT INTO** CONTAINS\_3 **VALUES (**6 , 104);

## Final Display:

1. **Floor**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| ID | MUSIC\_ID | FLOOR\_LEVEL | TOTAL\_SPACES | AVAILABLE\_SPACES | TOTAL\_DEPARTMENTS | Type |
| 0 | 10 | -3 | 0 | 0 | 0 | Parking Lot |
| 1 | 10 | -2 | 0 | 0 | 0 | Parking Lot |
| 2 | 9 | -1 | 0 | 0 | 0 | Parking Lot |
| 3 | 8 | 0 | 7 | 3 | 2 | Electronics |
| 4 | 7 | 1 | 9 | 6 | 2 | Clothes |
| 5 | 6 | 2 | 9 | 6 | 0 | Restaurants |
| 6 | 5 | 3 | 4 | 1 | 1 | Sports Equipment |

1. **Parking Lot**

|  |  |  |  |
| --- | --- | --- | --- |
| ID | Floor\_ID | Total\_Spaces | Available\_Spaces |
| 1 | 0 | 50 | 20 |
| 2 | 1 | 75 | 35 |
| 3 | 2 | 100 | 40 |

1. **Department**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| ID | FLOOR\_ID | MANAGER\_ID | YEARS\_OF\_EXPERIENCE | NAME | NUMBER\_OF\_EMPLOYEES |
| 0 | 3 | 1000 | 3 | Customer Service | 3 |
| 1 | 3 | 1003 | 4 | Security | 3 |
| 2 | 4 | 1006 | 5 | Accounting & Finance | 3 |
| 3 | 4 | 1009 | 3 | Promotions & Marketing | 2 |
| 4 | 6 | 1012 | 5 | Janitorial Services | 2 |

1. **Department\_Service\_Offered**

|  |  |
| --- | --- |
| Department\_ID | Service\_Offered |
| 0 | Customer Feedback Collection and Analysis |
| 0 | Answering client or customer questions |
| 1 | monitoring surveillance equipment |
| 1 | inspecting buildings, equipment, and access points |
| 1 | Crime prevention and detection |
| 2 | billing clients |
| 2 | managing payroll |
| 2 | Tracking assets and expenditures |
| 2 | Keeping track of critical tax documents. |
| 3 | Managing the Advertisements and public relations |
| 3 | Set the Spaces prices |
| 3 | Sponsor Events |
| 4 | Clean building floors |
| 4 | Operate on sections under maintenance |

1. **Employee**

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| ID | Department\_ID | FIRST\_NAME | MIDDLE\_INITIAL | LAST\_NAME | DATE\_OF\_BIRTH | GENDER | EMAIL | PHONE\_NUMBER | CITY | STREET\_NAME |
| 1000 | 0 | Charbel | P | Younis | 8/22/2000 | M | charbel.younis@lau.edu | 71448040 | Beirut | Badaro |
| 1001 | 0 | Georges | F | Freiha | 6/14/2000 | M | georges.freiha@lau.edu | 03351717 | Matn | Jdeide |
| 1002 | 0 | Bahaa | Y | Thebian | 1/1/2001 | M | bahaa.thebian@lau.edu | 79313543 | Baakline | Al-Wata |
| 1003 | 1 | Chris | M | Chalhoub | 2/2/2000 | M | chris.chalhoub@lau.edu | 70000180 | Matn | Mansourieh |
| 1004 | 1 | Jones | J | Johnson | 2/9/1997 | M | jones.johnson@shoppingmall.com | 03123456 | Beirut | Bliss |
| 1005 | 1 | Johnys | K | Freddy | 8/2/2001 | M | johnys.freddy@shoppingmall.com | 70123321 | beirut | downtown |
| 1006 | 2 | Sukwando | F | Smith | 10/4/1997 | M | sukwando.smith@shoppingmall.com | 80909090 | Matn | Bourj-Hammoud |
| 1007 | 2 | Lea | O | Abrahimis | 10/10/1999 | F | lea.abrahimis@shoppingmall.com | 71567768 | Beirut | Cornich |
| 1008 | 2 | Roberta | L | James | 10/1/2000 | F | roberta.james@shoppingmall.com | 78963014 | beirut | barbir |
| 1009 | 3 | Suksuk | M | Jordan | 1/18/1999 | M | suksuk.jordan@shoppingmall.com | 01377777 | beirut | hamra |
| 1010 | 3 | Imad | Y | mansour | 8/7/1999 | F | imad.mansour@shoppingmall.com | 71655990 | beirut | saray |
| 1011 | 4 | Jason | T | Paul | 1/10/2000 | M | jason.paul@shoppingmall.com | 03444534 | beirut | airport |
| 1012 | 4 | Moussa | R | Haidous | 9/9/2001 | M | moussa.haidous@lau.edu | 76696952 | beirut | hamra |

1. **Employement\_Contract**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| ID | Employee\_ID | START\_DATE | END\_DATE | SALARY | Role | DATE\_SIGNED |
| 1 | 1000 | 1/1/2020 | 1/1/2021 | 7000 | Customer Service Manager | 11/16/2020 |
| 2 | 1001 | 2/7/2020 | 2/7/2021 | 7000 | Customer Service Engineer | 11/16/2020 |
| 3 | 1002 | 3/6/2020 | 3/6/2022 | 7000 | Customer Service advisor | 11/16/2020 |
| 4 | 1003 | 11/24/2020 | 11/24/2021 | 7000 | Security Manager | 11/16/2020 |
| 5 | 1004 | 5/19/2020 | 5/19/2023 | 5000 | Security Agent | 11/17/2020 |
| 6 | 1005 | 7/18/2020 | 7/18/2021 | 4000 | Safety Advisor | 11/17/2020 |
| 7 | 1006 | 3/19/2020 | 3/19/2022 | 4000 | Accounting Manager | 11/17/2020 |
| 8 | 1007 | 5/1/2020 | 5/1/2021 | 4000 | **Financial Controller** | 11/18/2020 |
| 9 | 1008 | 9/25/2020 | 9/25/2021 | 4000 | Accountant | 11/18/2020 |
| 10 | 1009 | 7/13/2020 | 7/13/2021 | 6000 | Marketing Manager | 11/18/2020 |
| 11 | 1010 | 5/11/2020 | 5/11/2021 | 5500 | Marketing Consultant | 11/19/2020 |
| 12 | 1011 | 8/1/2020 | 8/12/2021 | 3000 | Janitorial Manager | 11/19/2020 |
| 13 | 1012 | 9/11/2020 | 9/11/2021 | 3700 | Janitorial Advisor | 11/19/2020 |

1. **Client**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| ID | BUSINESS\_NAME | PHONE\_NUMBER | EMAIL | Number\_OF\_RENTED\_SPACES | CITY | STREET\_NAME |
| 1 | ZARA | 01293949 | zaralebanon@gmail.com | 1 | beirut | kaskas street |
| 2 | MANGO | 01737482 | mangolebanon@gmail.com | 1 | beirut | hamra street |
| 3 | GS | 01555234 | gslebanon@outlook.com | 2 | beirut | makdessi street |
| 4 | AMERICAN EAGLE OUTFITTERS | 01290949 | aelebanon@gmail.com | 1 | beirut | barbir |
| 5 | PINKBERRY | 04123321 | pinkberrylebanon@gmail.com | 2 | beirut | verdun street |
| 6 | ATAMIAN WATCHES | 04555555 | atamianwatches @gmail.com | 0 | el metn | zalka street |
| 7 | FITNESS ZONE | 01293649 | fitnesszoneleb@gmail.com | 0 | beirut | hamra street |
| 8 | ZARA | 01293949 | zaralebanon@gmail.com | 1 | beirut | Kaskas street |
| 9 | AIZONE | 01123456 | aizoneleb@hotmail.com | 1 | Beirut | Jones street |
| 10 | ROMANCE TIMES | 01898989 | romancetimesleb@gmail.com | 1 | el beirut | al sanayeh street |
| 11 | MALAK EL TAWOUK | 01292929 | malakeltawouklebbanon@gmail.com | 1 | el metn | sagesse street |
| 12 | BURGER KING | 01333294 | bklebanon@gmail.com | 1 | Beirut | Kraytem |
| 13 | ROADSTER DINER | 01297777 | rdlebanon@hotmail.com | 0 | beirut | hamra street |

1. **Point of Contact**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| EMAIL | CLIENT\_ID | FIRST\_NAME | MIDDLE\_INITIAL | LAST\_NAME | PHONE\_NUMBER |
| joeyounis@gmail.com | 1 | Joe | C | Younis | 03493929 |
| charbelsuksuk@gmail.com, | 2 | Charbel | F | Suksuk | 71938483 |
| iambahas@gmail.com, | 3 | Bahas | H | Sukwando | 81229339 |
| dylanjones@gmail.com | 4 | Dylan | J | Jones | 03293920 |
| stonecold@gmail.com | 5 | Steve | C | Austin | 71442992 |
| andrewabb@gmail.com | 6 | Andrew | A | Abboud | 81332950 |
| alexandrekr@gmail.com | 7 | Alexandre | G | Krikorian | 70993848 |
| charbelhad@gmail.com | 8 | Charbel | P | Haddad | 03443921 |
| joerizk@gmail.com, | 9 | Joe | M | Rizk | 03473923 |
| charbelrah@gmail.com | 10 | Charbel | c | Rahal | 03455529 |
| touficsaade@gmail.com | 11 | Toufic | K | Saade | 03491629 |
| vanessa@gmail.com | 12 | Vanessa | N | Nunez | 71738489 |
| chrisbarakat@gmail.com | 13 | Christopher | G | Barakat | 71922999 |

1. **Spaces**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| ID | FLOOR\_ID | LENGTH | WIDTH | AVAILABLE | RENT\_COST | TYPE |
| 0 | 3 | 15 | 15 | Y | 3375 | Shop |
| 1 | 3 | 10 | 15 | N | 2250 | Restaurant |
| 2 | 3 | 10 | 8 | N | 800 | Shop |
| 3 | 3 | 20 | 25 | N | 5000 | Shop |
| 4 | 3 | 15 | 12 | N | 1800 | Shop |
| 5 | 3 | 3 | 2 | Y | 600 | Ad |
| 6 | 3 | 5 | 5 | N | 2500 | Ad |
| 7 | 4 | 10 | 8 | Y | 800 | Shop |
| 8 | 4 | 20 | 25 | Y | 5000 | Shop |
| 9 | 4 | 15 | 12 | N | 1800 | Shop |
| 10 | 4 | 10 | 8 | Y | 800 | Shop |
| 11 | 4 | 20 | 25 | Y | 5000 | Shop |
| 12 | 4 | 15 | 12 | N | 1800 | Shop |
| 13 | 4 | 15 | 12 | N | 1800 | Shop |
| 14 | 4 | 3 | 2 | Y | 600 | Ad |
| 15 | 4 | 5 | 5 | Y | 2500 | Ad |
| 16 | 5 | 10 | 8 | Y | 800 | Shop |
| 17 | 5 | 20 | 25 | N | 5000 | Shop |
| 18 | 5 | 15 | 12 | N | 1800 | Shop |
| 20 | 5 | 20 | 25 | N | 5000 | Shop |
| 21 | 5 | 15 | 12 | Y | 1800 | Shop |
| 22 | 5 | 3 | 2 | Y | 600 | Ad |
| 23 | 5 | 5 | 5 | Y | 2500 | Ad |
| 24 | 5 | 2 | 2 | Y | 400 | Ad |
| 25 | 6 | 15 | 15 | N | 3375 | Restaurant |
| 26 | 6 | 10 | 15 | N | 2250 | Restaurant |
| 27 | 6 | 10 | 5 | N | 1000 | Restaurant |
| 28 | 6 | 20 | 20 | N | 6000 | Restaurant |

1. **Rent Contracts**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| ID | Space\_ID | Point\_Of\_Contact\_Email | Client\_ID | Start\_Date | End\_Date | Agreed\_Rent\_Cost |
| 1 | 2 | joeyounis@gmail.com | 1 | 09-12-2021 | 09-12-2023 | 750 |
| 2 | 4 | charbelsuksuk@gmail.com | 2 | 10-12-2020 | 10-12-2022 | 1000 |
| 3 | 6 | iambahas@gmail.com | 3 | 05-12-2021 | 05-12-2024 | 2000 |
| 4 | 3 | dylanjones@gmail.com | 4 | 09-12-2021 | 09-12-2023 | 4500 |
| 5 | 12 | stonecold@gmail.com | 5 | 04-09-2021 | 04-12-2023 | 1750 |
| 6 | 9 | andrewabb@gmail.com | 6 | 02-01-2021 | 09-12-2023 | 1250 |
| 7 | 13 | alexandrekr@gmail.com | 7 | 05-12-2021 | 09-12-2024 | 1500 |
| 8 | 18 | charbelhad@gmail.com | 8 | 01-10-2021 | 09-12-2026 | 1500 |
| 9 | 17 | joerizk@gmail.com | 9 | 08-01-2021 | 08-01-2025 | 4250 |
| 10 | 20 | charbelrah@gmail.com | 10 | 09-12-2021 | 09-12-2023 | 5000 |
| 11 | 26 | touficsaade@gmail.com | 11 | 12-12-2018 | 03-12-2020 | 2250 |
| 12 | 25 | iambahas@gmail.com | 3 | 12-12-2018 | 12-27-2020 | 2250 |
| 13 | 1 | chrisbarakat@gmail.com | 13 | 12-12-2019 | 01-01-2020 | 2250 |

1. **Music**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ID | TITLE | COMPOSER | GENRE | DURATION |
| 1 | in my feelings | Drake | hip hop | 3:36 |
| 2 | Golden | Harry Styles | pop | 4:01 |
| 3 | are you with me | Lost Frequencies | electronic | 2:56 |
| 4 | a thousand times over | Ella Mai | R&B | 4:32 |
| 5 | Reality | Lost Frequencies | electronic | 3:09 |
| 6 | give a little | Maggie Rogers | Pop | 3:41 |
| 7 | In the end | Linkin Park | Rock | 3:04 |
| 8 | Fire on fire | Sam Smith | Pop | 5:49 |
| 9 | After hours | The Weekend | R&B | 6:17 |
| 10 | Imagine | John Lennon | Rock | 4:02 |

1. **Event**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ID | NAME | SPONSOR | START\_DATE | END\_DATE |
| 101 | Christmas | Mall shareholders | 11/28/2020 | 01/08/2021 |
| 102 | Halloween | Mall shareholders | 10/25/2020 | 11/01/2020 |
| 103 | Car Giveaway | Toyota car company | 4/4/2021 | 4/10/2021 |
| 104 | Red cross donation | Red Cross | 11/28/2020 | 01/08/2021 |
| 105 | Valentine | Mall Shareholders | 02/10/2021 | 02/18/2021 |
| 106 | Easter | Mall Shareholders | 04/12/2021 | 04/15/2021 |
| 107 | Black Friday | Mall Shareholders | 11/23/2021 | 11/28/2021 |
| 108 | Fashion week | Clothes Shops | 08/15/2021 | 08/23/2021 |
| 109 | Mothers Day | jewelery Shops | 03/18/2021 | 03/23/2021 |
| 110 | Perfume Testing | Lacoste Perfume Shop | 03/02/2021 | 03/03/2021 |

1. **Contains\_3**

|  |  |
| --- | --- |
| Floor\_ID | Event\_ID |
| 3 | 101 |
| 4 | 101 |
| 5 | 101 |
| 6 | 101 |
| 3 | 104 |
| 4 | 104 |
| 5 | 104 |
| 6 | 104 |

# 

# Sample Transactions

1-

**Statement:**

One of our potential sponsors would like to set up an event for christmas. They need at least 200 sqm of available shop space on the same floor, in addition to having a restaurant to service the customers. Moreover, they’d like to view our selection of pop music to be played across all floors. Query the database to ensure that we can satisfy their requirements.

**Solution:**

We’ll retrieve all the available shop spaces that have an area greater than 200 sqm that also have a rented restaurant space on the same floor. We’ll also grab the music with the pop genre for viewing by our client.

**Query:**

**SELECT** \* **FROM** SPACE

**WHERE** TYPE= 'Shop'

**AND** AVAILABLE= 'Y'

**AND** (WIDTH \* LENGTH>= 200 )

**AND** FLOOR\_ID IN

(

**SELECT DISTINCT** FLOOR\_ID **FROM** SPACE

**WHERE** TYPE= 'Restaurant'

**AND** AVAILABLE='N'

) ;

**Result:**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| ID | FLOOR\_ID | LENGTH | WIDTH | AVAILABLE | RENT\_COST | TYPE |
| 0 | 3 | 15 | 15 | Y | 3375 | Shop |

**Query:**

**SELECT** \*

**FROM** MUSIC

**WHERE** GENRE = 'Pop';

**Result:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ID | TITLE | COMPOSER | GENRE | DURATION |
| 6 | give a little | Maggie Rogers | Pop | 3:41 |
| 8 | Fire on fire | Sam Smith | Pop | 5:49 |

2-

**Statement:**

The Promotions and Marketing department is asking for $250000 as funding for a new ad campaign. The department expects the campaign to bring in 7 short-term clients (6 months - 12 months) and 3 long-term clients (3+ years). The Accounting and Finance” department will not grant the funding unless it can break even in 3 years. Query the database in such a way that it satisfies the above conditions with rent cost sorted in an decreasing order.

**Solution:**

To satisfy the above conditions, we’ll retrieve 10 of the highest rent cost available restaurant spaces and then the highest rent cost available shops for the Accounting and Finance department to do the required calculations.

**Query:**

**SELECT** \* **FROM** SPACE

**WHERE** TYPE = 'Restaurant' **AND** AVAILABLE = 'Y'  
 **UNION**  
 **SELECT** \* **FROM** SPACE

**WHERE** TYPE = 'Shop'

**AND** AVAILABLE = 'Y'

**ORDER BY** `rent\_cost` **DESC LIMIT** 10;

**Result:**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| ID | FLOOR\_ID | LENGTH | WIDTH | AVAILABLE | RENT\_COST | TYPE |
| 8 | 4 | 20 | 25 | Y | 5000 | Shop |
| 11 | 4 | 20 | 25 | Y | 5000 | Shop |
| 0 | 3 | 15 | 15 | Y | 3375 | Shop |
| 21 | 5 | 15 | 12 | Y | 1800 | Shop |
| 19 | 5 | 10 | 8 | Y | 800 | Shop |
| 16 | 5 | 10 | 8 | Y | 800 | Shop |
| 7 | 4 | 10 | 8 | Y | 800 | Shop |
| 10 | 4 | 10 | 8 | Y | 800 | Shop |

3-

**Statement:**

The next-gen consoles were just released and Microsoft & Sony are racing to place ads in our electronics floor. Microsoft is willing to rent all available ad spots on the electronics floor for 3 months in addition to at most a 100 sqm space on the same floor to set up stands and demo areas. Sony on the other hand, is willing to purchase all available ad spots in the entire mall in addition to 2 any-area available spaces for 1 month. Query the database and display in the data in such a way that it is obvious which company should the mall partner with.

**Solution:**

We’ll present both offers as SQL queries so that the total rent cost can be calculated easily and a company can be decided upon.

* **Microsoft’s Offer Query:**

**(SELECT** \*

**FROM** SPACE

**WHERE** TYPE= 'Ad'

**AND** AVAILABLE= 'Y'

**AND** FLOOR\_ID= 3)

UNION

**(SELECT** \*

**FROM** SPACE

**WHERE** TYPE = 'Shop'

**AND** FLOOR\_ID = 3

**AND** (WIDTH \* LENGTH<= 100 )

**AND** available = 'Y' );

**Result:**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| ID | FLOOR\_ID | LENGTH | WIDTH | AVAILABLE | RENT\_COST | TYPE |
| 5 | 3 | 3 | 2 | Y | 600 | Ad |

* **Sony’s Offer Query:**

**SELECT** \*

**FROM** SPACE

**WHERE** (type = 'Ad' OR type ='Shop')

**AND** AVAILABLE= 'Y'

**ORDER BY** RENT\_COST **DESC**;

**Result :**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| ID | FLOOR\_ID | LENGTH | WIDTH | AVAILABLE | RENT\_COST | TYPE |
| 8 | 4 | 20 | 25 | Y | 5000 | Shop |
| 11 | 4 | 20 | 25 | Y | 5000 | Shop |
| 0 | 3 | 15 | 15 | Y | 3375 | Shop |
| 15 | 4 | 5 | 5 | Y | 2500 | Ad |
| 23 | 5 | 5 | 5 | Y | 2500 | Ad |
| 21 | 5 | 15 | 12 | Y | 1800 | Shop |
| 16 | 5 | 10 | 8 | Y | 800 | Shop |
| 19 | 5 | 10 | 8 | Y | 800 | Shop |
| 7 | 4 | 10 | 8 | Y | 800 | Shop |
| 10 | 4 | 10 | 8 | Y | 800 | Shop |
| 22 | 5 | 3 | 2 | Y | 600 | Ad |
| 5 | 3 | 3 | 2 | Y | 600 | Ad |
| 14 | 4 | 3 | 2 | Y | 600 | Ad |
| 24 | 5 | 2 | 2 | Y | 400 | Ad |

It's clear that Sony is the one that the mall should partner with as it's more profitable.

4-

**Statement:**

Blackfriday is coming up and our major electronics store client would like to expand its store to accommodate the amount of people rushing to the mall. The client wants at least an additional 200 sqm of space on the same floor and is willing to buy out any client that has already rented the spaces. However, according to mall policy, clients are not allowed to sell their contracts if they expire within 6 months. Query the database such that it contains the available spaces for our client to rent.

**Solution:**

We’ll first grab all the available spaces on the 3rd floor with a minimum area of 200 sqm. We’ll then combine that result with all the spaces on the 3rd floor with a minimum area of 200 sqm and their corresponding rent contracts end after 6 months from the current date.

**Query:**

**( SELECT** \*

**FROM** SPACE

**WHERE** TYPE = ‘Shop’

**AND** AVAILABLE =’Y’

**AND** FLOOR\_ID = 3

**AND (**(WIDTH\*LENGTH)>=200))

**UNION**

**( SELECT** \*

**FROM** SPACE

**WHERE** ID **IN** ( **SELECT** SPACE\_ID **FROM** RENT\_CONTRACT  
 **WHERE** (  
 END\_DATE - SYSDATE > 180 )

**AND**  (WIDTH\*LENGTH)>=200));

**Result:**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| ID | FLOOR\_ID | LENGTH | WIDTH | AVAILABLE | RENT\_COST | TYPE |
| 0 | 3 | 15 | 15 | Y | 3375 | Shop |

5-

**Statement:**

Our security department has reported a breach into our Accounting and Finance department. The cameras were turned off so it must’ve been one of our own employees. However, the robber stupidly left a shredded piece of paper behind. It has on it the following information:

Salary : 5500

Phone Number: XX-XX59XX

City: XXXRUX

Help us find the culprit and fire him immediately!

**Solution:**

We will Determine the Employee\_ID who has a matching phone number and city using the inner query, and then see if they are bound to an Employment\_Contract with a salary 5,500. And fire him using the DELETE ( Employee and Employement\_Contract) operation

**Query:**

**SELECT**  \*

**FROM**  EMPLOYMENT\_CONTRACT

**WHERE** SALARY = 5500 **AND** EMPLOYEE\_ID =

( **SELECT** ID

**FROM** EMPLOYEE

**WHERE** PHONE\_NUMVER **LIKE** '\_\_\_\_59%'

**AND** CITY **LIKE** '\_\_\_ru\_' ) ;

**Result:**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| ID | EMPLOYEE\_ID | START\_DATE | END\_DATE | SALARY | ROLE | DATE\_SIGNED |
| 11 | 1010 | 5/11/2020 | 5/11/2021 | 5500 | Marketing Consultant | 11/19/2020 |

**DELETE FROM** EMPLOYMENT\_CONTRACT

**WHERE** EMPLOYEE\_ID=1010;

**DELETE FROM** EMPLOYEE

**WHERE** ID=1010;

The Employee has been fired and accordingly deleted from the database!

6-

**Statement:**

One of our potential clients came to us after a successful marketing campaign looking for spaces to rent. However, as the client was coming up to our offices, they saw two previous employees that used to work for them, “Georges Freiha” and “Bahaa Thebian” and apparently their relationship has been sour lately. We do not usually accommodate personal quarrels such as this one but this client might rent for long term and hence we’ll try to please them. They want all spaces such that “Georges Freiha” and “Bahaa Thebian” do not work in a department on the same floor as the spaces they hope to rent.

**Solution:**

We’ll get all the spaces that are on floors that do not contain the departments that the above two employees work at.

**Query:**

**SELECT** \*

**FROM** SPACE

**WHERE** AVAILABLE ='Y' AND NOT FLOOR\_ID =

(

**SELECT** FLOOR\_ID

**FROM** DEPARTMENT

**WHERE** ID= (SELECT DEPARMTENT\_ID **FROM** EMPLOYEE

**WHERE** ( (FIRST\_NAME = 'Georges' **AND** LAST\_NAME = 'Freiha') **OR** (FIRST\_NAME = 'Bahaa' **AND** LAST\_NAME = 'Thebian'))));

**Result:**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| ID | FLOOR\_ID | LENGTH | WIDTH | AVAILABLE | RENT\_COST | TYPE |
| 7 | 4 | 10 | 8 | Y | 800 | Shop |
| 8 | 4 | 20 | 25 | Y | 5000 | Shop |
| 10 | 4 | 10 | 8 | Y | 800 | Shop |
| 11 | 4 | 20 | 25 | Y | 5000 | Shop |
| 14 | 4 | 3 | 2 | Y | 600 | Ad |
| 15 | 4 | 5 | 5 | Y | 2500 | Ad |
| 16 | 5 | 10 | 8 | Y | 800 | Shop |
| 19 | 5 | 10 | 8 | Y | 800 | Shop |
| 21 | 5 | 15 | 12 | Y | 1800 | Shop |
| 22 | 5 | 3 | 2 | Y | 600 | Ad |
| 23 | 5 | 5 | 5 | Y | 2500 | Ad |
| 24 | 5 | 2 | 2 | Y | 400 | Ad |

7-

**Statement:**

The Accounting and Finance department wants to check all the rent contracts that will expire within the next 3 months so that they can negotiate contract renewals. Query the database to show the results and order them according to the agreed rent cost in decreasing order.

**Solution:**

We’ll subtract the END\_DATE from the current date to query all the rent contracts that end during the next 3 months (90 days)

**Query:**

**SELECT** \*

**FROM** RENT\_CONTRACT

**WHERE** END\_DATE - SYSDATE<=90 )

**ORDER BY** rent\_cost **DESC**;

**Result:**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| ID | Space\_ID | Point\_Of\_Contact\_Email | Client\_ID | Start\_Date | End\_Date | Agreed\_Rent\_Cost |
| 12 | 25 | iambahas@gmail.com | 3 | 12-12-2018 | 12-27-2020 | 2250 |
| 13 | 1 | chrisbarakat@gmail.com | 13 | 12-12-2019 | 01-01-2020 | 2250 |

8-

**Statement:**

A potential client wants to rent a space with at most 300 sqm of area and at most rent cost of 2,000 on the Clothes floor. For the opening, he wants to sponsor an event. However, he wants to make sure no other event is currently active. Get all the possible spaces for our client and show him all the events scheduled for the next month so that they could plan on a suitable time slot.

**Solution:**

**Query:**

* Possible Spaces

**SELECT** \*

**FROM** SPACE

**WHERE** AVAILABLE='Y'

**AND** (WIDTH \* LENGTH <= 300)

**AND** (RENT\_COST <= 2000)

**AND** FLOOR\_ID = (**SELECT**  ID **FROM** FLOOR **WHERE** TYPE= 'Clothes');

**Result:**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| ID | FLOOR\_ID | LENGTH | WIDTH | AVAILABLE | RENT\_COST | TYPE |
| 7 | 4 | 10 | 8 | Y | 800 | Shop |
| 10 | 4 | 10 | 8 | Y | 800 | Shop |
| 14 | 4 | 3 | 2 | Y | 600 | Ad |

* Possible Time Slots

**Query:**

**SELECT** \*

**FROM** EVENT

**WHERE** START\_DATE - SYSDATE <= 90 )

**AND** END\_DATE - SYSDATE <= 3);

**Results:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ID | NAME | SPONSOR | START\_DATE | END\_DATE |
| 101 | Christmas | Mall shareholders | 11/28/2020 | 01/08/2021 |
| 104 | Red cross donation | Red Cross | 11/28/2020 | 01/08/2021 |

9-

**Statement:**

We received an email from “allegedly” one of our clients asking for us to terminate their rent contract immediately. However, we have a great relationship with this client specifically and we suspect the email we received from “[dylannjones@gmail.com](mailto:dylannjones@gmail.com)” is fraudulent. Please double check if this email is indeed in our point of contacts table and corresponds to our client “AMERICAN EAGLE OUTFITTERS”. Also grab us our Security department manager’s phone number.

**Solution:**

* Check if fraud:

**Query:**

**SELECT** \*

**FROM** POINT\_OF\_CONTACT

**WHERE** EMAIL = 'dylannjones@gmail.com'

**AND** CLIENT\_ID = (**SELECT**  ID **FROM** CLIENT **WHERE** BUSINESS\_NAME = 'AMERICAN EAGLE OUTFITTERS');

**Result:**

No Data was found

**He is a fraud!!!!!!!**

* Get Security department manager’s phone number

**Query:**

**SELECT** PHONE\_NUMBER

**FROM** EMPLOYEE

**WHERE** ID = ( **SELECT**  MANAGER\_ID

**FROM** DEPARTMENT

**WHERE** NAME = 'Security');

**Result:**

70000180

10-

**Statement:**

Anticipating the dollar crisis in Lebanon, our Accounting and Finance hid a “Increase agreed rent cost at any time” clause in all our rent contracts that Starts after 01-01-2021. Although unfortunate, the time has come for us to raise our rent. Please grab us all the rent contracts that have been signed after we inserted our clause and increase their rent by 10%

**Solution:**

We will use a function that will determine the contracts that have been signed after 01-01-2021 and increase the agreed\_rent\_cost accordingly.

**Query:**

**UPDATE** `rent\_contract`

**SET** AGREED\_RENT\_COST = AGREED\_RENT\_COST \* 1.10

**WHERE** START\_DATE >= TO\_DATE('2021-01-01', 'yyyy-mm-dd' );

**Result:**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| ID | Space\_ID | Point\_Of\_Contact\_Email | Client\_ID | Start\_Date | End\_Date | Agreed\_Rent\_Cost |
| 1 | 2 | joeyounis@gmail.com | 1 | 09-12-2021 | 09-12-2023 | 825 |
| 3 | 6 | iambahas@gmail.com | 3 | 05-12-2021 | 05-12-2024 | 2200 |
| 4 | 3 | dylanjones@gmail.com | 4 | 09-12-2021 | 09-12-2023 | 4950 |
| 5 | 12 | stonecold@gmail.com | 5 | 04-09-2021 | 04-12-2023 | 1925 |
| 6 | 9 | andrewabb@gmail.com | 6 | 02-01-2021 | 09-12-2023 | 1375 |
| 7 | 13 | alexandrekr@gmail.com | 7 | 05-12-2021 | 09-12-2024 | 1650 |
| 8 | 18 | charbelhad@gmail.com | 8 | 01-10-2021 | 09-12-2026 | 1650 |
| 9 | 17 | joerizk@gmail.com | 9 | 08-01-2021 | 08-01-2025 | 4675 |
| 10 | 20 | charbelrah@gmail.com | 10 | 09-12-2021 | 09-12-2023 | 5500 |

# VII- Normalization Up to The BCNF Normal Form:

After creating all relations we should improve them by normalizing according to several normal forms. Here we are going to normalize our database up to the fourth normal form which is the BoyceCodd Normal Form. In each relation we are going to apply the four normal forms. We start with the first then second then third and at last the BCNF normal form. Let us first start by a general description to each normal form.

## First Normal Form:

This form does not allow multivalued attributes, composite attributes, and their combinations to exist in a relation.

1. Only attribute values permitted are single atomic values.

2. Domain of an attribute must only include atomic values and the value of an attribute in a tuple

must be a single value from the domain of that attribute.

3. Disallows having a set of values as an attribute value for a single tuple.

## Second Normal Form:

The Second normal form is based on the concept of full functional dependency. Before explaining the second form let us define some concepts used in this form and other forms also.

**Functional Dependencies:** A constraint between two sets of attributes from the database. The values of the Y component of a tuple in relation R depend on, or are determined by the values of an X component. We say that Y is functionally dependent on X.

**Prime attribute:** An attribute that is a member of the primary key, all other attributes are called non-prime attributes

**Full functional dependency:** A functional dependency X →Y is a full functional dependency if removal of any attribute A from X means that the dependency does not hold anymore.

**Partial Dependency:** A functional dependency X →Y is a partial functional dependency if removal of any attribute A from X means that the dependency still holds.

A relation schema R is in the second normal form if it satisfies both of the following conditions

1. It satisfies the conditions of the first normal form
2. every non-prime attribute in R is fully functionally dependent on the primary key of R and every non-prime attribute A in R is not partially dependent on the primary key in R.

## Third Normal Form:

A relation schema R is in the third normal form if it satisfies both of the following conditions

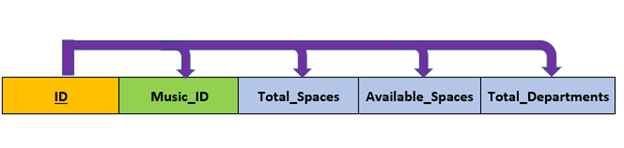
1. It satisfies the conditions of the second normal form
2. There exists no functional dependency X→A where X is not a super key or A is not a prime attribute

## Boyce Codd Normal Form:

The Boyce Codd normal form is a stricter form than the third normal form. The BCNF differs from the definition of the third normal form in only one condition. The third normal form allows the right hand side of the functional dependency to be a prime attribute while BCNF does not allow that.

A relation is said to be on BCNF if it satisfy both of the following Conditions

1. It satisfies the conditions of the Third normal form
2. There exists no functional dependency X→A where X is not a super key.
3. **Floor :**



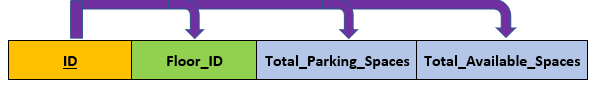
**A** . The **Floor** relation schema satisfies all conditions of the 1NF because it has neither multivalued attributes nor composite attributes. All attributes are single and atomic.

**B**. The **Floor** relation schema satisfies all conditions of the 2NF because every non-prime attribute is fully functionally dependent on the primary key “ID”.

**C**. The **Floor** relation schema satisfies all conditions of the 3NF because there exists no functional dependency X→A where X is not a super key or A is not a prime attribute

**D**. The **Floor** relation schema satisfies all conditions of the BCNF because there exists no functional dependency X→A where X is not a super key.

1. **Parking\_Lot :**



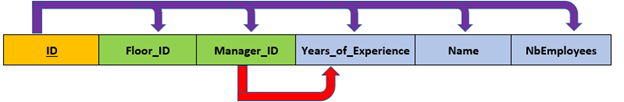
**A.** The **Parking\_lot** relation schema satisfies all conditions of the 1NF because it has neither multivalued attributes nor composite attributes. All attributes are single and atomic.

**B.** The **Parking\_lot** relation schema satisfies all conditions of the 2NF because every non-prime attribute is fully functionally dependent on the primary key **“ID”**.

**C.** The **Parking\_lot** relation schema satisfies all conditions of the 3NF because there exists no functional dependency X→A where X is not a super key or A is not a prime attribute

**D.** The **Parking\_lot** relation schema satisfies all conditions of the BCNF because there exists no functional dependency X→A where X is not a super key

1. **Department :**



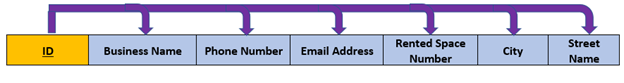
**A.** The **Department** relation schema satisfies all conditions of the 1NF because it has neither multivalued attributes nor composite attributes. All attributes are single and atomic.

**B.** The **Department** relation schema satisfies all conditions of the 2NF because every non-prime attribute is fully functionally dependent on the primary key **“ID”**.

**C.** The **Department** relation schema satisfies all conditions of the 3NF because there exists no functional dependency X→A where X is not a super key or A is not a prime attribute

**D.** The **Department** relation schema satisfies all conditions of the BCNF because there exists no functional dependency X→A where X is not a super key

1. **Client :**



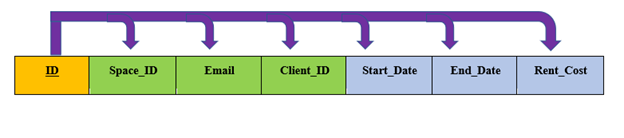
**A.** The **Client** relation schema satisfies all conditions of the 1NF because it has neither multivalued attributes nor composite attributes. All attributes are single and atomic.

**B.** The **Client** relation schema satisfies all conditions of the 2NF because every non-prime attribute is fully functionally dependent on the primary key **“ID”**.

**C.** The **Client** relation schema satisfies all conditions of the 3NF because there exists no functional dependency X→A where X is not a super key or A is not a prime attribute.

**D.** The **Client** relation schema satisfies all conditions of the BCNF because there exists no functional dependency X→A where X is not a super key.

1. **Rent\_contract :**

****

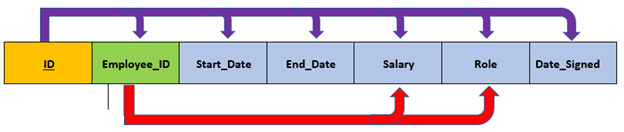
**A**. The **Rent\_contract** relation schema satisfies all conditions of the 1NF because it has neither multivalued attributes nor composite attributes. All attributes are single and atomic.

**B**. The **Rent\_contract** relation schema satisfies all conditions of the 2NF because every non prime attribute is fully functionally dependent on the primary key “ID”.

**C.** The **Rent\_contract** relation schema satisfies all conditions of the 3NF because there exists no functional dependency X→A where X is not a super key or A is not a prime attribute.

**D.** The **Rent\_contract** relation schema satisfies all conditions of the BCNF because there exists no functional dependency X→A where X is not a super key

1. **Employement\_Contract :**

****

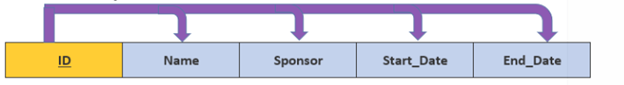
**A**. The **Employement\_Contract** relation schema satisfies all conditions of the 1NF because it has neither multivalued attributes nor composite attributes. All attributes are single and atomic.

**B**. The **Employement\_Contract** relation schema satisfies all conditions of the 2NF because every non-prime attribute is fully functionally dependent on the primary key “ID”.

**C.** The **Employement\_Contract** relation schema satisfies all conditions of the 3NF because there exists no functional dependency X→A where X is not a super key or A is not a prime attribute.

**D**. The **Employement\_Contract** relation schema satisfies all conditions of the BCNF because there exists no functional dependency X→A where X is not a super key.

1. **Event :**

****

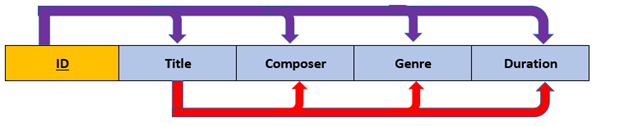
**A**. The **Event** relation schema satisfies all conditions of the 1NF because it has neither multivalued attributes nor composite attributes. All attributes are single and atomic.

**B**. The **Event** relation schema satisfies all conditions of the 2NF because every non-prime attribute is fully functionally dependent on the primary key “ID”.

**C.** The **Event** relation schema satisfies all conditions of the 3NF because there exists no functional dependency X→A where X is not a super key or A is not a prime attribute.

**D**. The **Event** relation schema satisfies all conditions of the BCNF because there exists no functional dependency X→A where X is not a super key.

1. **Music :**

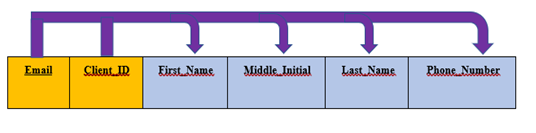


**A.** The **Music** relation schema satisfies all conditions of the 1NF because it has neither multivalued attributes nor composite attributes. All attributes are single and atomic.

**B.** The **Music** relation schema satisfies all conditions of the 2NF because every non-prime attribute is fully functionally dependent on the primary key **“ID”**.

**C.** The **Music** relation schema satisfies all conditions of the 3NF because there exists no functional dependency X→A where X is not a super key or A is not a prime attribute.

**D**. The **Music** relation schema satisfies all conditions of the BCNF because there exists no functional dependency X→A where X is not a super key.

1. **Point of contact : **

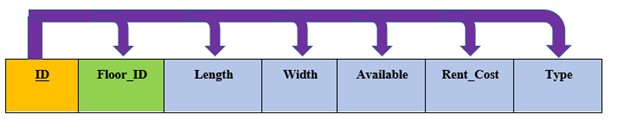
**A.** The **POINT\_OF\_CONTACT** relation schema satisfies all conditions of the 1NF because it has neither multivalued attributes nor composite attributes. All attributes are single and atomic.

**B.** The **POINT\_OF\_CONTACT** relation schema satisfies all conditions of the 2NF because every non prime attribute is fully functionally dependent on the primary key “**EMAIL+CLIENT\_ID**”.

**C.** The **POINT\_OF\_CONTACT** relation schema satisfies all conditions of the 3NF because there exists no functional dependency X→A where X is not a super key or A is not a prime attribute.

**D**. The **POINT\_OF\_CONTACT** relation schema satisfies all conditions of the BCNF because there exists no functional dependency X→A where X is not a super key.

1. **Space :**

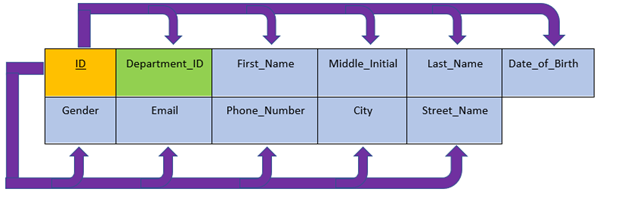
****

**A**. The **Space** relation schema satisfies all conditions of the 1NF because it has neither multivalued attributes nor composite attributes. All attributes are single and atomic.

**B**. The **Space** relation schema satisfies all conditions of the 2NF because every non prime attribute is fully functionally dependent on the primary key “ID”.

**C.** The **Space** relation schema satisfies all conditions of the 3NF because there exists no functional dependency X→A where X is not a super key or A is not a prime attribute.

**D**. The **Space** relation schema satisfies all conditions of the BCNF because there exists no functional dependency X→A where X is not a super key.



**A.** The **Employee** relation schema satisfies all conditions of the 1NF because it has neither multivalued attributes nor composite attributes. All attributes are single and atomic.

**B.** The **Employee** relation schema satisfies all conditions of the 2NF because every non prime attribute is fully functionally dependent on the primary key **“ID”**.

**C.** The **Employee** relation schema satisfies all conditions of the 3NF because there exists no functional dependency X→A where X is not a super key or A is not a prime attribute.

**D**. The **Employee** relation schema satisfies all conditions of the BCNF because there exists no functional dependency X→A where X is not a super key.

**Relation Schemas without non-prime attributes:**

|  |  |
| --- | --- |
| Department\_ID | Service\_Offered |

|  |  |
| --- | --- |
| Floor\_ID | Event\_ID |

# VIII - Conclusion

All in all, designing this database was a great experience for all of us. Not only did we learn the true power of group work (this project could not have been done alone) but we learned more about the work ethic required to design and create good-quality databases.

Our final database design is far from perfect but it definitely is our best work yet. After multiple iterations, arguments and discussions, and long hours of meetings and coordination, we hope we have presented a database design that is of great quality.

Thank you Dr. Haraty for this great opportunity.