****

**DATABSE DESIGN PROJECT FOR SPORTS CLUB**

**BY**

**HADY HAMMOUD**

**YOUSSEF HOTEIT**

**“CMPS342”: Data Base Management Systems**

**In Computer Science**

**SOFTWARE DEVELOPMENT TEAM NAME: HY**

**A black and white logo

Description automatically generated**

**April 20, 2024**

**Abstract**

This report provides a comprehensive overview of the database application system of the sports club. The database design was implemented using the Entity-Relationship Model to demonstrate the relationship between entities and attributes. Also, it provides detailed information regarding each entity, its associated attributes, and the relationship between different entities.

The database architecture serves as a foundational model for implementing a functional database system. The significance of this system is its ability to potentially optimize data storage, retrieval, and management of the sports club. It will also explore how the conceptional design will be transformed and mapped to logical design.

**I. Table of contents**

II. Introduction……………………………………………………………………… 4

III. System Requirements…………………………………………………………… 5

IV. ER-Diagram…………………………………………………………………...... 15

V. Flowchart Symbols………………………………………………………………. 16

VI. Relations…………………………...................................................................... 21

VII. Table Creation………………………………………………………………… 27

VIII. Table Description ……………………………………………………………… 35

IX. Queries………………………………………………………………………… 41

X. Conclusion……………………………………………………………………. 57

**II. Introduction**

This database report serves as an initiative to design a database for Barcelona Sports Club. The collaborative team, known as HY and comprised of Hadi and Yousef is dedicated to tailoring a database to meet the sports club's requirements.

The first part of the project will concentrate on developing a comprehensive understanding of the attributes and entities constituting the database application system. This phase also involves visually depicting the relationships between various entities through the illustration of the Entity-Relationship Model (ER-Model).

The second part of the project focuses on transitioning the conceptual design into a logical structure. This transition is accomplished through the application of a seven-step mapping algorithm, systematically mapping regular entities, weak entities, relationships, and addressing multi-valued attributes using foreign keys and partial keys.

The subsequent parts objective is to utilize SQL to create tables, visualize them, insert data, and execute various queries. The final phase will involve data normalization. From start to completion, the project spans the abstract phase through the concrete implementation.

**III. System Requirements**

A sports club serves as a hub for athletes of diverse nationalities engaging in various sports, for both male and female players, with athletes being the central focus.

Staff members within the club acquire specific roles ensuring operational smoothness and communication across the club's venues. These roles convey: coaches responsible for athlete training, managers handling administrative aspects, medical staff ensuring athlete well-being, and administrative personnel managing operational functions like hiring staff and overseeing venue operations.

The club houses multiple venues, including stadiums for games, gyms for training, cafeterias, and an infirmary. Each athlete possesses an individual profile, indicating personal details, medical history, blood type, and physical fitness records essential for their well-being.

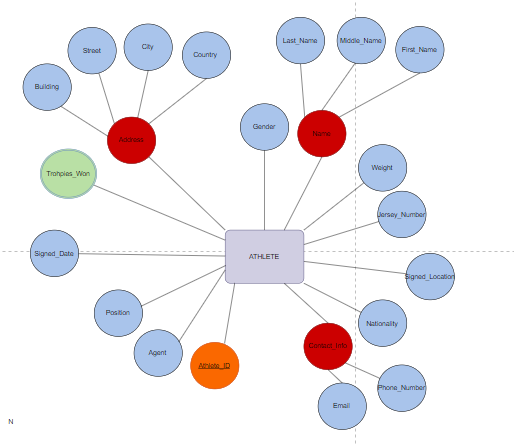
Coaches conduct tailored training sessions at various club locations, consisting of diverse drills targeting specific muscle groups and utilizing various equipment. Athletes participate in matches organized by sport-specific teams, each assigned a unique ID and distinct leagues, with game statistics recording key moments like goals, penalties, and injuries during play.

The champions of each game are awarded a trophy, each denoting the win's location, the opposing team, and the victory date.

Every club member, including athletes, coaches, and managerial staff, signs individual contracts that vary based on roles, detailing start and end dates, renewal terms, bonuses, and specific clauses, such as early termination clauses for athletes.

The support club system is composed of ten strong entities which are: Athlete, Staff , Venue , Team , Training\_Session , Training\_Drill , Trophy , Match ,Contract and two weak entities Medical\_Record and Match\_Stastics.

1. ***ATHLETE:***

******

ID**:** it is unique identification number for each player whereby the first three letters indicate the type of sport for example BBP for basket

ball player followed by three sequential numbers like 001.

Name: a composite attribute made up of Athlete’s First name, Middle name, Last name

Contact\_Info: a composite attribute having two key attributes: email and phone\_number that indicats the phone number and email address of the athlete.

Gender: defines the gender of the athlete Male or Female

Nationality: indicates where the player is from

Trophy\_Won: a multi-valued attribute that indicates the awards won by each player.

Address: is a composite attribute made of country of residence, and city.

Jersey\_Number: indicates the jersey number worn by the athlete.

Agent: indicates the name of the agent responsible for the athlete’s professional affairs.

Position: indicates the position of the athlete in their sport.

1. ***MEDICAL\_RECORD:***

A diagram of medical record

Description automatically generated

Date: a partial key that refers to the athlete corresponding to the medical record.

Previous injuries*:* indicates the type of injuries the player had.

Medication*:* multi-values attribute indicates the medications used by each player

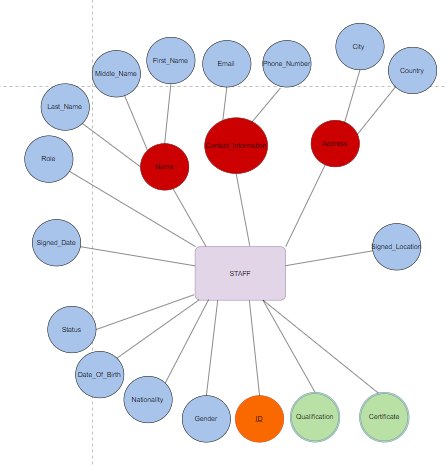
Height: the height of the player in cm.

Weight: the weight of the player in kg.

Blood\_Type: indicates the blood type of the athlete if it is A+/-, B+/-, O+/-, AB+/-

Date\_of\_Birth: indicates the birthdate of the athlete.

1. ***STAFF:***

******

ID**:** it is a unique nine-digit identification number for each employee whereby the first four digits indicate year of joining the club, second two digits indicate the department the employee works in.

Name: a composite attribute made of the employee's first name, middle name, and last name.

Address: is a composite attribute made of country of residence, city.

Contact\_Info: a composite attribute having two key attributes: email and phone\_number that indicats the

Qualification: multi-valued attribute indicating the eligibility requirements of the employee.

Certificate*:* multi-valued attribute indicating certificates for each employee.

Gender: defines the gender of the staff Male or Female

Nationality: indicates where the staff is from

Date\_of\_birth: indicates the date of birth of the staff.

Status: indicates the current employment status of the staff member (active, on vacation, resigned…)

Role: indicates the role of the staff member.

1. ***VENUE:***

A diagram of a venue

Description automatically generated

Name: a key attribute that is unique to each building and it indicates the name of the building

ID: a one-digit key attribute that identifies each building

Location: an attribute that identifies location of each building in the club

Capacity: an attribute that indicates the maximum number of participants in each building

Contact\_Info: a composite attribute that indicated the email, and phone number of the person in charge of the building.

Type: indicates the type of venue (e.g. stadium, arena, cafeteria ..)

Description: indicates the service provided by each building

1. ***TEAM:***

***A diagram of a team

Description automatically generated***

League Name**:** a key attribute that indicates the name of the team.

ID: a key attribute that identifies each team

Sport\_Type: a simple attribute that identifies the sport type each games plays.

Standings: a simple attribute that indicates the ranking of each team.

Number\_of\_Players: a simple attribute that indicates the number of players in each team.

1. ***TRAINING\_SESSION:***

A diagram of training session

Description automatically generated

ID**:** a key attribute that indicates the ID of the training session.

Location: is an attribute that indicates where the training session was held

Duration: indicates the duration of the session

Date: indicates the date of the training session.

1. ***TRAINING\_DRLL:***

A diagram of training drill

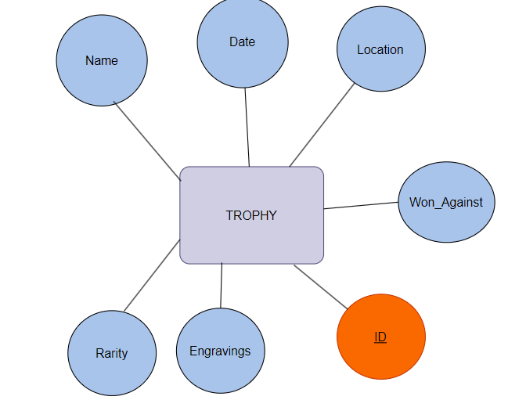
Description automatically generated

ID: key attribute identifying each training drill

Name: a key attribute identifying each drill

Difficulty\_Level: a scale from 0-10 that indicates the level of each drill.

1. ***TROPHY:***



ID**:** a six-digit identification number that indicates the id of each trophy whereby the first four digits indicate the year when the trophy was received, and the last two digits indicate the type of sport this trophy belongs to.

Date: indicates when the award was won.

Won\_Against: indicates the name of the opposing team.

Location: indicates the name of the stadium where the award was won

Name:indicates the name of the trophy

Rarirty: indicates the rarity of the trophy (common, rare, unique)

Engravings: indicates if any special engravings on the trophy

1. ***MATCH:***

A diagram of a match

Description automatically generated

ID**:** key attribute that indicates the id of each game.

Date: it indicates the date of the game as timestamp

Opponent\_Team: the name of the visiting team

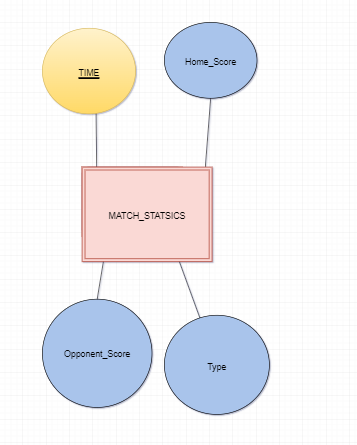
Stadium: indicates the name of the stadium where the game took place.

Match\_Type: indicates the type of the match (friendly, charity, or league match …)

Referees: a multivalue attributes that displays the referees of the match

Attendance: indicates the number of people who attended the game.

1. ***MATCH\_STATSTICS:***

******

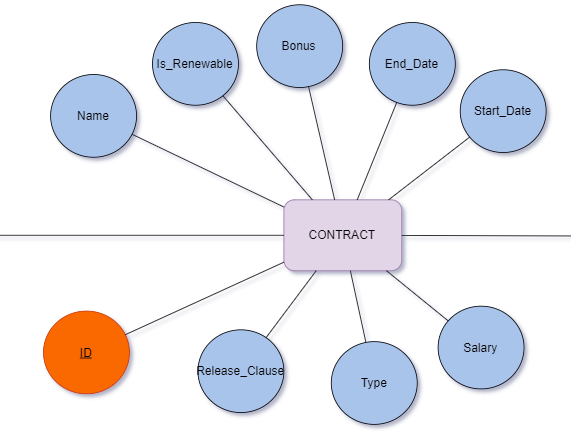
Time: partial key that indicates when did the action occur. Example: at time 90 minutes a goal was scored.

Home\_Score: indicates the score of the home team

Opponent\_Score:indicates the score of the opponent team

Type: indicates the type of this statistics (e.g., offside, 3pts, goal …)

1. ***CONTRACT***



ID: a key attribute that indicates the id of each contract.

Name: indicates the name of the contract

Salary: indicates the annual salary for the employee

Type: categorizes the type of contract as athlete contract or employee contract

End\_Date: indicates the termination date of the contract.

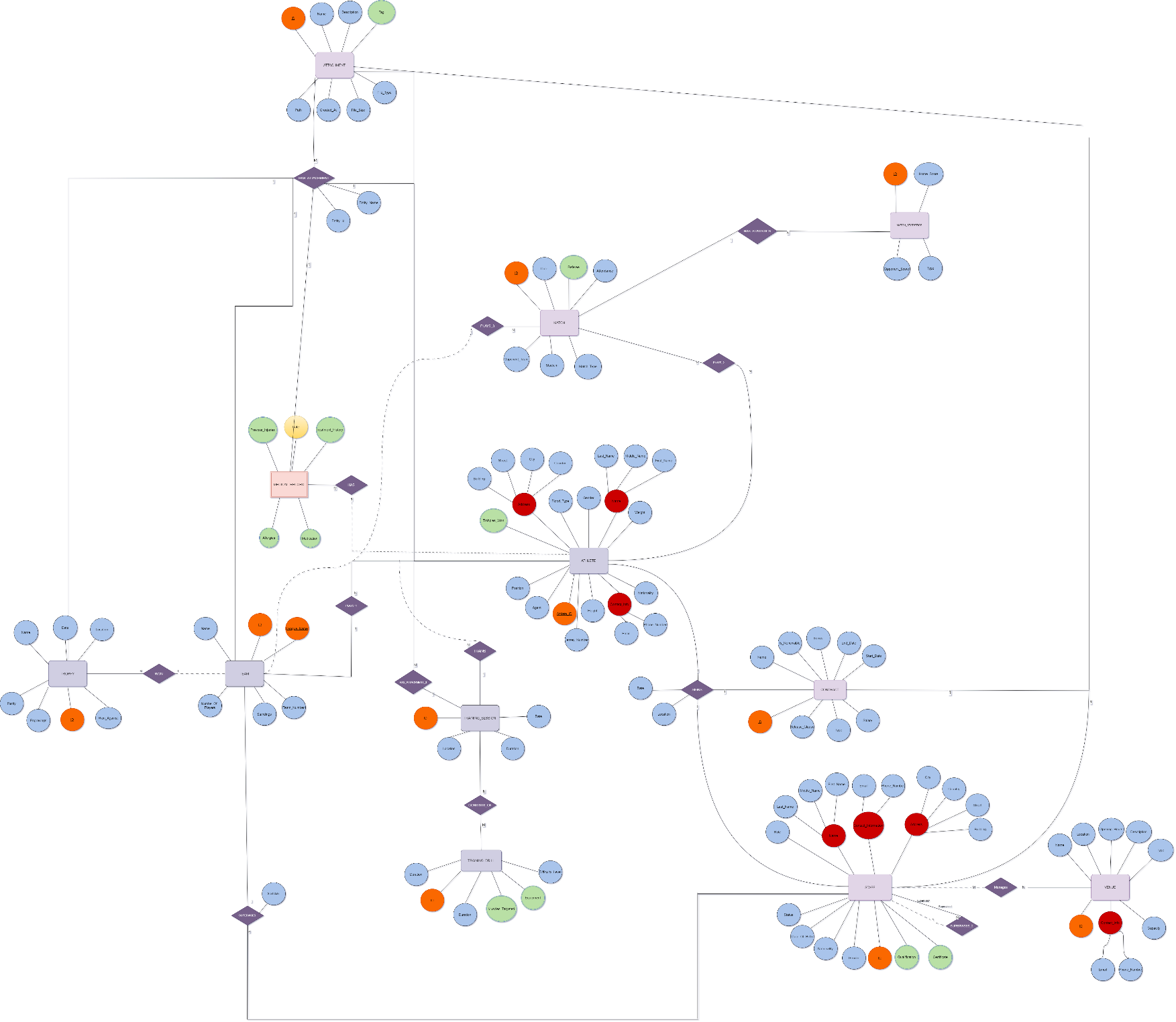
Is\_Renewable: indicates whether the contract is extendable, subject to renewal or not

Bonus: indicates the percentage of bonus added (if applicable)

Start\_Date indicates the commencement date of the contract.

Release\_clause: indicates any conditions, terms, or clauses related to early termination or renewal of the contract.

**IV. ER Diagram**



**V. Flowchart Symbols**

|  |  |
| --- | --- |
| Entity Name | A close-up of a purple card  Description automatically generated |
| Weak Entity | A pink rectangular sign with black text  Description automatically generated |
| Key Attribute | An orange oval with black text  Description automatically generated |
| Partial Key | A yellow circle with black text  Description automatically generated |
| Composite Attribute | A red oval with black text  Description automatically generated |
| Multi Valued Attribute | A green oval with black text  Description automatically generated |
| Simple Attribute |  |
| Relationship | A purple diamond with white text  Description automatically generated |
| Total Participation | A black background with a black square  Description automatically generated with medium confidence |
| Partial Participation | A black background with a black square  Description automatically generated with medium confidence |

**VI. Relations**

After creating the ER schema and describing the database for the Sports Club in terms of entities, attributes, and relationships, the conceptual design will be translated into a logical design using mapping algorithms. The algorithm consists of seven steps to map regular entities, weak entities, binary 1:1, binary 1:M, binary N: N, and N-array relationships, each of which will be described in detail.

**Step one: Mapping of regular entity types:**

For each regular entity type, a relation (table) is created. The key attribute of the entity becomes the primary key (which is underlined) and contains all the simple attributes of the entity. In the case of a composite attribute, the individual simple attributes constituting it will be included in the table. The regular entities to be mapped are as follows:

1. ATHLETE

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| ID | First\_Name | Middle\_Name | Last\_Name | Email | Phone\_Number | City |
| Country | Nationality | Agent | Position | Jersey\_Number | Gender |  |

   The Athlete entity contains a Primary key ID and simple attributes.The simple attributes are : First\_Name , Middle\_Name , Last Name , Gender , Country , City , Nationality , Email, Phone\_Number ,Jersey\_Number , Agent , Position.

2.STAFF

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| ID | First\_Name | Middle\_Name | Last\_Name | Gender | Nationality |
| Country | City | Email | Phone\_Number | Status | Role |

 The Staff entity contains ID as primary key and simple attributes that are: First\_Name , Middle\_Name , Last\_Name, Country , City , Email , Phone\_Number,Status , Date\_Of\_Birth, Status ,Nationalility , Gender, and Role.

3.VENUE

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| ID | Name | Location | Description | Type | Capacity | Email | Phone\_Number |

  The Venue entity contains an ID primary key and four simple attributes. The simple attributes are Location, Email, Phone\_Number , Capacity, Type , Description.

4.TRAINING\_SESSION

|  |  |  |  |
| --- | --- | --- | --- |
| ID | Location | Date | Duration |

This entity has ID as primary key and Location, Date, and Duration as simple attributes.

5.TRAINING\_DRILL

|  |  |  |
| --- | --- | --- |
| ID | Name | Difficulty\_Level |

The Training\_Drill entity has two simple attributes: Name and Difficulty\_Level and a primary key ID.

6.TROPHY

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| ID | Date | Location | Won\_Against | Rarity | Engravings | Name |

    The Trophy entity has one primary key which is the Trophy\_ID including four simple attributes which are: Date, Location, Won\_Against.

7.MATCH

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| ID | Date | Opponent\_Team | Stadium | Stadium | Match\_Type | Attendance |

The Match entity has ID as a primary key and simple attribute which are: Date, Opponent\_Team, Stadium, Attdendance, Match\_Type, Stadium.

8. CONTRACT

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| ID | Type | Bonus | Salary | Release\_Clause | Name | Start\_Date | End\_Date | Renewable |

The Contract entity consists of one primary key attribute which is the ID,and  eight simple attributes which are: Type, Bonus, Salary, Release\_Clause, Name, Start\_Date, End\_Date, and Renewable.

**Step two: Mapping of weak entity types:**

The weak entity type is mapped to a relation. The primary key of the weak entity type comprises a combination of both the primary key of the entity it refers to and its partial key. Multi-valued attributes are excluded from the table (they will be addressed later). In this database, there are two weak entities:

Medical\_Record and Match\_Statstics:

1. MEDICAL\_RECORD

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Athlete\_ID | DATE | Height | Weight | Blood\_Type | Date\_of\_Birth |

The Entity Type contains 2 multi-valued attributes which are not included in this relation. (they will be mapped later in step 6) and four simple attributes: Height, Weight, Blood\_Type and Date\_of\_Birth. The simple attribute Date is the partial key. The primary key of the owner entity renamed as Athlete\_ID and DATE make up the primary key of Medical Record.

1. MATCH\_STATSTICS

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| ATHLETE\_ID | MATCH\_ID | TIME | Type | Opponent\_Score | Home\_Score |

Match\_Statstics has three simple attributes Type, Opponent\_Score, Home\_Score and a partial key Time. The primary keys of Match and Athlete are added as foreign keys since both are owner entities. The primary key of Match\_Statstics is the combination of ATHLETE\_ID , MATCH\_ID , TIME.

**Step Three: Mapping of 1:1 Binary relationships:**

For mapping a 1:1 relationship, no new table (relation) is created. Instead, one of the participating entities is chosen (preferably the entity with total participation) to include the primary key of the other participating entity. This method is known as the foreign key approach, where the primary attribute of the other entity is added to the entity with total participation. In the case of total participation from both sides, either entity can be selected to incorporate the foreign key into its relation. In this database design there are two 1:1 relationship to be mapped.

1. ATHLETE (Signs)

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| ID | First\_Name | Middle\_Name | Last\_Name | Email | Phone\_Number | City | Gender |
| Country | Nationality | Agent | Position | Jersey\_Number | Contract\_ID | Sign\_Date | Sign\_Location |

Since both entities ATHLETE and CONTRACT have total participation, the foreign key can be added to either entity. In this case, the “Athlete” entity will be used for mapping. Every athlete signs a contract. The signs relationship links both entities, and the foreign key, which is primary key of “Contract”, was added to the “Athlete” relationship. The primary key of Contract “ID” is renamed to “Contract\_ID’. Sign\_Date and Sign\_Location are simple attributes for the relationship signs.

1. STAFF (Signs)

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| ID | First\_Name | Middle\_Name | Last\_Name | Email | Phone\_Number | City | Gender |
| Country | Nationality | Status | Role | Date\_Of\_Birth | Contract\_ID | Sign\_Date | Sign\_Location |

Since both entities STAFF and CONTRACT have total participation, the foreign key can be added to either entity. In this case, the “Staff” entity will be used for mapping. Every athlete signs a contract. The signs relationship links both entities, and the foreign key, which is primary key of “Contract”, was added to the “Staff” relationship. The primary key of Contract “ID” is renamed to “Contract\_ID’. Sign\_Date and Sign\_Location are simple attributes for the relationship signs.

**Step four: Mapping 1: N relationships:**

For mapping 1: N relationships, a new relation is not created. Instead, the primary key of the entity type from the 'one' side is added to the relation of the entity on the 'many' sides. This relation will contain the simple attributes of the one-to-many relationships. The one-to-many relationships requiring mapping are:

1. STAFF(SUPERVISES)

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| ID | First\_  Name | Middle\_Name | Last\_Name | Country | City | Start\_Date |
| Street | Building | Email | Phone\_Number | Role | Team\_ID |  |

Entity Type Relationship Staff SUPERVISES Team. The ‘SUPERVISES’ relationship links the 'STAFF' entity with the 'TEAM' entity. The 'STAFF' entity is on the 'Many' sides, which necessitates adding the foreign key 'ID' to its relationship. This foreign key is named ‘Team\_ID’ ’ and refers to the primary key of the 'TEAM' entity.Start\_Date is a simple attribute for the relationship SUPERVISES.

1. STAFF(SUPERVISES\_2)

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| ID | First\_  Name | Middle\_Name | Last\_Name | Country | City |
|  | Building | Email | Phone\_Number | Role | STAFF\_ID |

Entity Type Relationship Staff(supervisee) SUPERVISES\_2 a Staff (supervisor) which is a recursive relationship. The 'SUPERVISES\_2 ‘SUPERVISES\_2’relationship links the 'Staff' entity with itself. The ‘Staff(supervise)’ entity is on the 'Many' sides, which necessitates adding the foreign key 'ID' to its relationship. This foreign key is named ‘Staff\_ID’ and refers to the primary key of the 'Staff' entity.

Entity Type Relationship Athlete HAS a Medical Record. The ‘HAS’ relationship links the 'ATHLETE' entity with the MEDICAL RECORD ' entity. The ‘MEDICAL RECORD’ entity is on the 'Many' sides, which necessitates adding the foreign key 'ID' to its relationship. This foreign key is named ‘Athlete\_ID’ and refers to the primary key of Athlete.

1. ATHLETE(TRAINS)

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| ID | First\_Name | Middle\_Name | Last\_Name | Gender | Country | City |  |
| Street | Building | Nationality | Phone\_Number | Height | Weight | Blood\_Type | TRAINING\_ID |

Entity Type Relationship Athlete TRAINS at a Training\_Session. The ‘TRAINS’ relationship links the ‘ATHLETE’ entity with the ‘TRAINING SESSION' entity. The ‘ATHLETE’ entity is on the 'Many' sides, which necessitates adding the foreign key 'ID' to its relationship. This foreign key is named ‘TRAINING\_ID’ and refers to the primary key of the ‘TRAINING SESSION' ' entity.

1. ATHLETE(PLAYS\_1)

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| ID | First\_Name | Middle\_Name | Last\_Name | Gender | Country | City |  |
| Street | Building | Nationality | Phone\_Number | Height | Weight | Blood\_Type | TEAM\_ID |

Entity Type Relationship Athlete PLAYS\_1 in a Team. The ‘PLAYS\_1’relationship links the ‘ATHLETE’ entity with the 'TEAM' entity. The ‘ATHLETE’ entity is on the 'Many' sides, which necessitates adding the foreign key 'ID' to its relationship. This foreign key is named ‘Team\_ID’ and refers to the primary key of the 'TEAM' entity.

1. MATCH(PLAYS\_3)

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| ID | Date | Opponent\_Team | Stadium | Audience | TEAM\_ID |

Entity Type Relationship TEAM PLAYS\_3 in a MATCH. The ‘PLAYS\_3’relationship links the ‘MATCH’ entity with the 'TEAM' entity. The ‘MATCH’ entity is on the 'Many' sides, which necessitates adding the foreign key 'ID' to its relationship. This foreign key is named ‘Team\_ID’ and refers to the primary key of the 'TEAM' entity.

1. TEAM(WINS)

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| ID | Date | Location | Won\_Against | Rarity | Engravings | Name | TEAM\_ID |

Entity Type Relationship TEAM WON TROPHY. The ‘WON’ relationship links the 'TROPHY' entity with the ‘TEAM’ entity. The ‘TROPHY’ entity is on the 'Many' sides, which necessitates adding the foreign key 'ID' to its relationship. This foreign key is named ‘TEAM\_ID’ and refers to the primary key of the ‘TEAM’ entity.

1. MATCH(HAS\_STATICTICS)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ID | Opponent\_Score | Home\_Score | Type | Match\_ID |

Entity Type Relationship Match HAS\_STATICTICS Match\_Statistics. The ‘HAS\_STATICTICS relationship links the ''Match\_Statistics” entity with the ‘Match’ entity. The ‘’Match\_Statistics’ entity is on the 'Many' sides, which necessitates adding the foreign key 'ID' to its relationship. This foreign key is named ‘Match\_ID’ and refers to the primary key of the ‘Match’ entity.

**Step five: Mapping N:M relationships:**

For mapping M: N relationships, a new relationship is created. This new relation will contain, as foreign keys, the primary keys of the participating relations. The combination of both foreign keys serves as the primary key of the newly created relation. If the many-to-many relationship includes any simple attributes, these are also included in the newly created relationship. The many-to-many relationships that need to be mapped are MANAGES, PLAYS\_2, CONSISTS\_OF:

1. **MANAGES**

|  |  |
| --- | --- |
| Staff\_ID | Venue\_ID |

Many Staff members manage many Venues. The ‘MANAGES’ relationship links the two entity types: 'Staff' and 'Venue’. The ‘MANAGES’ relationship includes the primary keys of both entities. The primary key of the 'Staff' entity, originally named 'ID' is now 'Staff\_ID' Similarly, the primary key of the 'Venue' entity, initially named 'ID' is now ‘Venue\_ID’. Both newly named primary keys are added to the ‘MANAGES’ relationship, and the combination of both serves as the primary key for the ‘MANAGES’ relationship.

1. **PLAYS\_2**

|  |  |
| --- | --- |
| Athlete\_ID | Match\_ID |

Many Athletes Compete in Many Matches. The ‘MATCH\_PARTICIPATION’ relationship links the two entity types: ‘Athlete’ and 'Match.' The 'MATCH\_PARTICIPATION' relation includes the primary keys of both entities. The primary key of the 'Athlete' entity, originally named 'ID' is now 'Athlete\_ID'. Similarly, the primary key of the 'Match' entity, initially named ID’ is now ‘Match\_ID' Both newly named primary keys are added to the ‘PLAYS\_2’ relation, and the combination of both serves as the primary key for the ‘PLAYS\_2’ relation.

1. **CONSISITS\_OF:**

|  |  |
| --- | --- |
| Training\_session\_ID | Training\_drill\_ID |

Many Training\_sessions Consist\_of Many Training\_drills. The “CONSISTS\_OF” relayionship links the two entity types it is renamed to “TRAINING\_SESSION\_DRILL”: “Training\_session” and “Training\_drill”. The primary key of the 'Traning\_session' entity, originally named 'ID' is now ‘Training\_session\_ID’. Similarly, the primary key of the ' Training\_drill ' entity, initially named ‘ID’ is now ‘Training\_drill\_ID '. The combination of both primary keys makes up the primary key of new relation “CONSISTS\_OF”.

**Step six: Mapping of multivalued attributes:**

To map multivalued attributes, a new relation is created. This relation will include the primary key of the owning entity along with the related attribute. The combination of both serves as the primary key to the new relation. The multivalued attributes in this database that need to be mapped are:

**Athlete:**

Trophy\_Won (Athlete\_Trophies\_Won)

|  |  |
| --- | --- |
| Athlete\_ID | Trophy\_Won |

The multivalued attribute “Trophy\_Won” belongs to Athlete entity. To represent the attribute, a new relation is created. It demonstrates the multiples trophies an athlete can win. Athlete\_ID, the renamed primary key of the Athlete entity, and the attribute Trophy\_won compose the primary key of this relation.

**Staff:**

1. Certificate (Staff\_Certificate)

|  |  |
| --- | --- |
| Staff\_ID | Certificate |

The multivalued attribute “Certificate” belongs to the Staff entity. To represent the attribute, a new relation is created. It represents the multiple certificates a staff member can have. Staff\_ID the renamed primary key of Staff entity, and the attribute Certificate compose the primary key of this relation.

1. Qualification:(Staff\_Qualification)

|  |  |
| --- | --- |
| Staff\_ID | Qualification |

The multivalued attribute “Qualifications” belongs to the Staff entity. To represent the attribute, a new relation is created. It represents the multiple qualifications a staff member can possess. Staff\_ID the renamed primary key of Staff entity, and the attribute Qualification together compose the primary key of this relation.

**Medical Record:**

1. Previous\_Injury (Medical\_Record\_Previous\_Injury)

|  |  |  |
| --- | --- | --- |
| Athlete\_ID | Date | Previous\_Injusry |

The multivalued attribute “Previous Injuries” belongs to the weak entity Medical\_Record. To represent the attribute, a new relation is created. It represents (if any) the date and the description of the previous injuries of each athlete. The primary key of Medical\_Record entity (Athlete\_ID , Date) and the attribute Previous\_Injries compose the primary key of this relation.

|  |  |  |
| --- | --- | --- |
| Athlete\_ID | Date | Medication |

1. Medication (Medical\_Record\_Medication)

The multivalued attribute “Medication” belongs to the weak entity Medical\_Record. To represent the attribute, a new relation is created. It provides (if any) the medications prescribed for each athlete. The primary key of Medical\_Record entity (Athlete\_ID, Date) and the attribute Medication compose the primary key of this relation.

**Match:**

Referee (Match\_Refrree):

|  |  |
| --- | --- |
| Match\_ID | Referee |

The multivalued attribute “Referee” belongs to Match entity. To represent the attribute, a new relation is created. It indicates info regarding the referees of each match. The primary key of Match entity (Match\_ID) and the attribute Referee compose the primary key of this relation.

**Step seven: Mapping of N-array relationships:**

For mapping N-array relationships, a new relation is created. The primary key of this relation consists of a combination of the foreign keys representing the participating entity types from all participating relations. Additionally, this relation will include any simple attributes that the relationship possesses. Notably, in this database design, there is one Ternary relationship.

1. Participates\_Training\_Session:

|  |  |  |
| --- | --- | --- |
| TRAINING\_SESSION\_ID | ATHLETE\_ID | TEAM\_ID |

The primary key of each Athlete, Team, Training\_Session is added to the relation. The combination of those serves as the Primary key of Training\_Session\_Participation.

1. Participates\_in\_Match:

|  |  |  |
| --- | --- | --- |
| MATCH\_ID | ATHLETE\_ID | TEAM\_ID |

The primary key of each Athlete, Team, Match is added to the relation. The combination of those serves as the Primary key of Participates\_in\_Match:

**VII.Table Creation**

After mapping the entities, relationships and multi-valued attributes using the 7-steps of mapping algorithm, this stage of the report aims to create them using SQL.

1. Athlete:

CREATE TABLE "ATHLETE"

(

"ID" VARCHAR(6) PRIMARY KEY,

"First\_Name" VARCHAR(40) NOT NULL,

"Middle\_Name" VARCHAR(40) NOT NULL,

"Last\_Name" VARCHAR(40) NOT NULL,

"Gender" VARCHAR(1) NOT NULL CHECK("Gender" IN ('F','M')),

"Country" VARCHAR(40) NOT NULL,

"City" VARCHAR(40) NOT NULL,

"Nationality" VARCHAR(40) NOT NULL,

"Email" VARCHAR(40) UNIQUE NOT NULL,

"Phone\_Number" VARCHAR(40) UNIQUE NOT NULL,

"Jersey\_Number" INT NOT NULL,

"Agent" VARCHAR(40) NOT NULL,

"Position" VARCHAR(40) NOT NULL,

"Contract\_ID" VARCHAR(5) NOT NULL,

"Team\_ID" INT NOT NULL,

"Signed\_Location" VARCHAR(50) NOT NULL ,

"Signed\_Date" DATE NOT NULL,

CONSTRAINT "FK\_Contract\_ID" FOREIGN KEY ("Contract\_ID") REFERENCES "CONTRACT"("ID"),

CONSTRAINT "FK\_Team\_ID" FOREIGN KEY ("Team\_ID") REFERENCES "TEAM"("ID"),

CONSTRAINT "UQ\_Jersey\_Team" UNIQUE ("Jersey\_Number", "Team\_ID")

);

1. Staff:

Note: The club is based in Spain, but hires only Lebanese, Syrian, Jordanian, and Palestinian staff members.

CREATE TABLE "STAFF" (

"ID" VARCHAR(6) PRIMARY KEY,

"First\_Name" VARCHAR(40) NOT NULL,

"Middle\_Name" VARCHAR(40),

"Last\_Name" VARCHAR(40) NOT NULL,

"Gender" VARCHAR(1) NOT NULL CHECK ("Gender" IN ('M', 'F')),

"Country" VARCHAR(40) NOT NULL,

"City" VARCHAR(40) NOT NULL,

"Email" VARCHAR(40) NOT NULL UNIQUE,

"Role" VARCHAR(40) NOT NULL,

"Phone\_Number" VARCHAR(40) NOT NULL CHECK (REGEXP\_LIKE("Phone\_Number", '^\+[0-9]+-[0-9]+$')),

"Status" VARCHAR(20) NOT NULL CHECK ("Status" IN ('Active', 'Inactive', 'On Vacation','Fired')),

"Date\_Of\_Birth" DATE NOT NULL,

"Nationality" VARCHAR(20) NOT NULL CHECK ("Nationality" IN ('Lebanon', 'Syria', 'Jordan', 'Palestine')),

"Start\_Date" Date NOT NULL,

"TEAM\_ID" NUMBER,

"Contract\_ID" VARCHAR(6) NOT NULL,

FOREIGN KEY("Contract\_ID") REFERENCES "CONTRACT"("ID"),

FOREIGN KEY ("TEAM\_ID") REFERENCES "TEAM"("ID")

);

CREATE OR REPLACE TRIGGER check\_age

BEFORE INSERT OR UPDATE ON "STAFF"

FOR EACH ROW

DECLARE

v\_age NUMBER;

BEGIN

v\_age := TRUNC(MONTHS\_BETWEEN(SYSDATE, :NEW."Date\_Of\_Birth") / 12);

IF v\_age < 18 THEN

RAISE\_APPLICATION\_ERROR(-20001, 'Age must be 18 or older');

END IF;

END;

ALTER TABLE "STAFF"

ADD ("Supervisor\_ID" VARCHAR(6));

ALTER TABLE "STAFF"

ADD CONSTRAINT "FK\_SUPERVISOR\_STAFF"

FOREIGN KEY ("Supervisor\_ID")

REFERENCES "STAFF"("ID");

ALTER TABLE "STAFF" ADD("Start\_Date" Date NOT NULL)

ALTER TABLE "STAFF" ADD("Signed\_Date" Date )

ALTER TABLE "STAFF" ADD("Signed\_Location" VARCHAR(50) )

1. Venue:

CREATE TABLE "VENUE" (

"ID" NUMBER PRIMARY KEY,

"Location" VARCHAR2(100),

"Capacity" NUMBER CHECK ("Capacity" > 0),

"Email" VARCHAR2(40) UNIQUE CHECK (REGEXP\_LIKE("Email", '^[A-Za-z0-9.\_%-]+@[A-Za-z0-9.-]+\.[A-Z|a-z]{2,4}$')),

"Phone\_Number" VARCHAR2(40) UNIQUE NOT NULL CHECK (REGEXP\_LIKE("Phone\_Number", '^\+[0-9]+-[0-9]+$')),

"Description" VARCHAR2(255) NOT NULL,

"Type" VARCHAR2(255) NOT NULL CHECK ("Type" IN ('Cafeteria', 'Parking Lot', 'Training Stadium', 'Official Stadium', 'Health Building')),

"Name" VARCHAR2(255) NOT NULL

);

1. Team:

CREATE TABLE "TEAM" (

"ID" NUMBER NOT NULL,

"League\_Name" VARCHAR(50) UNIQUE NOT NULL,

"Name" Varchar(50) NOT NULL,

"Sport\_Type" VARCHAR(17) NOT NULL,

"Standings" NUMBER NOT NULL CHECK("Standings" > 0 AND "Standings" <30),

"Number\_of\_Players" NUMBER ,

PRIMARY KEY ("ID")

);

1. Training\_Session:

CREATE TABLE “TRAINING\_SESSION"

( "ID" NUMBER PRIMARY KEY,

"Location" VARCHAR2(50) NOT NULL,

"Date" TIMESTAMP NOT NULL,

"Duration" NUMBER NOT NULL

);

1. Training\_Drill:

CREATE TABLE "TRAINING\_DRILL"(

"ID" NUMBER Primary key NOT NULL,

"Name" VARCHAR2(20) UNIQUE NOT NULL,

"Description" VARCHAR2(255) NOT NULL,

"Difficulty\_Level" VARCHAR2(255) NOT NULL,

"Duration" NUMBER NOT NULL

);

1. Trophy:

CREATE TABLE "TROPHY" (

"ID" VARCHAR(5) PRIMARY KEY NOT NULL,

"Date" DATE NOT NULL,

"Won\_Against" CHAR(50) NOT NULL CHECK (LENGTH("Won\_Against") > 3),

"Rarity" VARCHAR(6) NOT NULL CHECK ("Rarity" IN ('Common', 'Rare', 'Unique')),

"Engravings" VARCHAR(50) ,

"Location" VARCHAR(50) DEFAULT 'Barcelona' NOT NULL,

"Team\_ID" INT NOT NULL,

"Name" VARCHAR(50) NOT NULL CHECK (LENGTH("Name") > 3),

FOREIGN KEY ("Team\_ID") REFERENCES "TEAM"("ID")

);

1. Match:

CREATE TABLE "MATCH" (

"ID" VARCHAR(5) PRIMARY KEY,

"DATE" DATE NOT NULL,

"Opponent\_Team" VARCHAR(40) NOT NULL,

"Stadium" VARCHAR(50) NOT NULL,

"Attendance" INT NOT NULL CHECK ("Attendance" >= 0),

"Match\_Type" VARCHAR(40) NOT NULL CHECK ("Match\_Type" IN ('Friendly','Charity','League','Competition')) ,

"TEAM\_ID" INT NOT NULL,

FOREIGN KEY ("TEAM\_ID") REFERENCES TEAM("ID")

);

1. Contract:

CREATE TABLE "CONTRACT" (

"ID" VARCHAR2(5) PRIMARY KEY,

"Type" VARCHAR(40) NOT NULL,

"Bonus" NUMBER NOT NULL CHECK ("Bonus" >= 0),

"Salary" NUMBER NOT NULL CHECK ("Salary" >= 0),

"Release\_Clause" NUMBER NOT NULL CHECK ("Release\_Clause" >= 0),

"Start\_Date" DATE NOT NULL,

"End\_Date" DATE NOT NULL ,

"Renewable" NUMBER(1) NOT NULL CHECK ("Renewable" IN (0, 1)),

CHECK ("End\_Date" > "Start\_Date")

);

1. Medical\_Record:

CREATE TABLE "MEDICAL\_RECORD" (

"Date" DATE,

"Athlete\_ID" VARCHAR(6),

"Height" VARCHAR(30),

"Weight" VARCHAR(30),

"Blood\_Type" VARCHAR(5),

PRIMARY KEY ("Date" , "Athlete\_ID"),

FOREIGN KEY ("Athlete\_ID") REFERENCES ATHLETE("ID")

);

1. Match\_Statstics:

CREATE TABLE "MATCH\_STATISTICS"(

"Time" Int NOT NULL,

"Athlete\_ID" VARCHAR(6) NOT NULL,

"Match\_ID" VARCHAR(5) NOT NULL,

"Type" VARCHAR(20) NOT NULL,

PRIMARY KEY("Time","Athlete\_ID","Match\_ID"),

FOREIGN KEY ("Match\_ID") REFERENCES MATCH("ID") ,

FOREIGN KEY ("Athlete\_ID") REFERENCES ATHLETE("ID")

)

1. Match\_Participation:

CREATE TABLE "MATCH\_PARTICIPATION"(

"Match\_ID" VARCHAR(5) NOT NULL,

"Athlete\_ID" VARCHAR(6) NOT NULL ,

"Team\_ID" INT NOT NULL,

PRIMARY KEY("Match\_ID","Athlete\_ID","Team\_ID"),

FOREIGN KEY ("Match\_ID") REFERENCES "MATCH"("ID"),

FOREIGN KEY ("Athlete\_ID") REFERENCES "ATHLETE"("ID"),

FOREIGN KEY ("Team\_ID") REFERENCES "TEAM"("ID")

)

1. Manages:

CREATE TABLE "MANAGES"(

"Venue\_ID" VARCHAR(6) NOT NULL,

"Staff\_ID" VARCHAR(6) NOT NULL,

PRIMARY KEY("Venue\_ID","Staff\_ID"),

FOREIGN KEY ("Venue\_ID") REFERENCES "VENUE"("ID"),

FOREIGN KEY ("Staff\_ID") REFERENCES "STAFF"("ID")

)

1. Consists\_Of

CREATE TABLE "CONSISTS\_OF" (

"Session\_ID" NUMBER,

"Drill\_ID" VARCHAR2(5),

PRIMARY KEY ("Session\_ID", "Drill\_ID"),

FOREIGN KEY ("Session\_ID") REFERENCES "TRAINING\_SESSION"("ID"),

FOREIGN KEY ("Drill\_ID") REFERENCES "TRAINING\_DRILL"("Drill\_ID")

);

1. Training\_Session\_Participation:

CREATE TABLE "TRAINING\_SESSION\_PARTICIPATION"(

"Training\_Session\_ID" INT NOT NULL,

"Athlete\_ID" VARCHAR(6) NOT NULL ,

"Team\_ID" INT NOT NULL,

PRIMARY KEY("Training\_Session\_ID","Athlete\_ID","Team\_ID"),

FOREIGN KEY ("Training\_Session\_ID") REFERENCES "TRAINING\_SESSION"("ID"),

FOREIGN KEY ("Athlete\_ID") REFERENCES "ATHLETE"("ID"),

FOREIGN KEY ("Team\_ID") REFERENCES "TEAM"("ID")

)

1. Staff\_Qualification:

CREATE TABLE "STAFF\_QUALIFICATION" (

"Staff\_ID" VARCHAR(40) NOT NULL,

"Degree" VARCHAR(50) NOT NULL,

"Year\_Completed" NUMBER(4) NOT NULL,

Primary Key("Staff\_ID","Degree"),

FOREIGN KEY ("Staff\_ID") REFERENCES "STAFF"("ID")

)

1. Staff\_Certificate:

CREATE TABLE "STAFF\_CERTIFICATE" (

"Staff\_ID" VARCHAR(6) NOT NULL,

"Certificate\_Name" VARCHAR(50) NOT NULL,

"Issue\_Date" DATE NOT NULL

"Expiry\_Date" DATE,

Primary Key("Staff\_ID","Certificate\_Name"),

FOREIGN KEY ("Staff\_ID") REFERENCES "STAFF"("ID"),

CHECK("Issue\_Date" < "Expiry\_Date")

)

1. Medical\_Record\_Previous\_Injury:

CREATE TABLE "MEDICAL\_RECORD\_PREVIOUS\_INJURY"(

"Athlete\_ID" VARCHAR(6) NOT NULL,

"Date" Date NOT NULL,

"Injury" VARCHAR(255) NOT NULL,

Primary key ("Athlete\_ID","Date","Injury"),

FOREIGN KEY ("Athlete\_ID") REFERENCES "ATHLETE"("ID")

)

1. Medical\_Record\_Medication:

CREATE TABLE "MEDICAL\_RECORD\_MEDICATION"(

"Athlete\_ID" VARCHAR(6) NOT NULL,

"Date" Date NOT NULL,

"Medication" VARCHAR(255) NOT NULL,

Primary key ("Athlete\_ID","Date","Medication")

)

1. Athlete\_Trophies\_Won:

CREATE TABLE "ATHLETE\_TROPHIES\_WON"(

"Athlete\_ID" VARCHAR(6) NOT NULL,

"Trophy" VARCHAR(50) NOT NULL,

"Date" Date NOT NULL,

Primary key("Athlete\_ID","Trophy"),

FOREIGN KEY ("Athlete\_ID") REFERENCES "ATHLETE"("ID")

)

1. Match\_Referee:

CREATE TABLE "MATCH\_REFEREE"(

"Match\_ID" VARCHAR(5) NOT NULL,

"Referee" VARCHAR(50) NOT NULL,

"Rating" Number(4,2) NOT NULL CHECK("Rating">=0.0 AND "Rating"<=10.00) ,

PRIMARY KEY( "Match\_ID","Referee"),

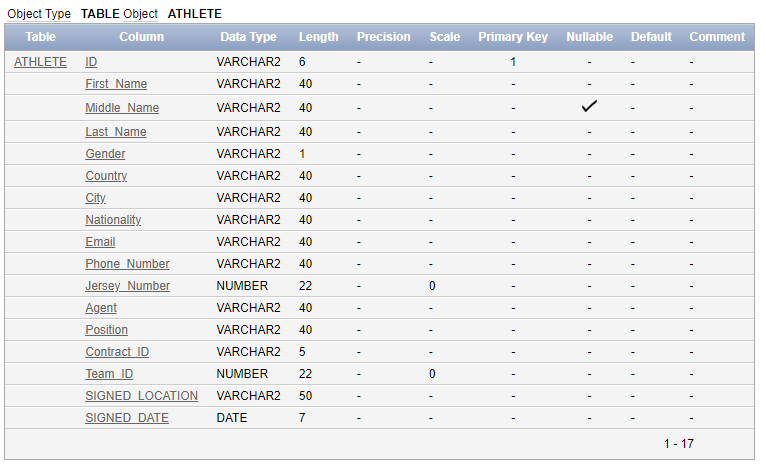
FOREIGN KEY ("Match\_ID") REFERENCES MATCH("ID")

)

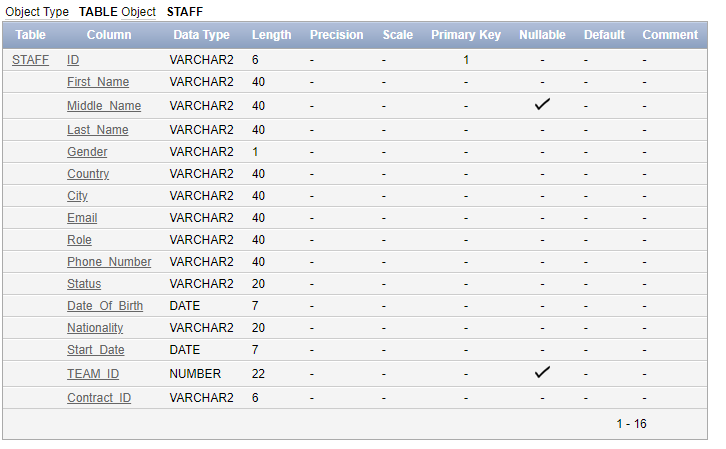
**VIII. Table Description**

After creating the tables, we can now have a view on the relations.

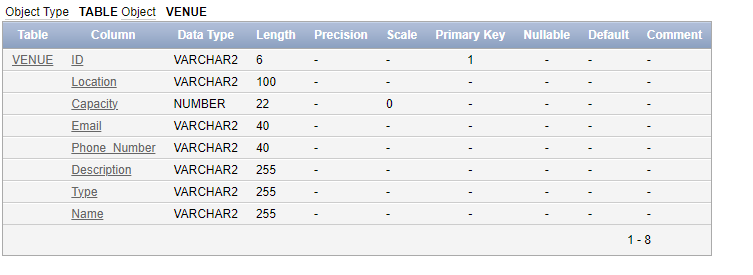
1. DESC ATHLETE



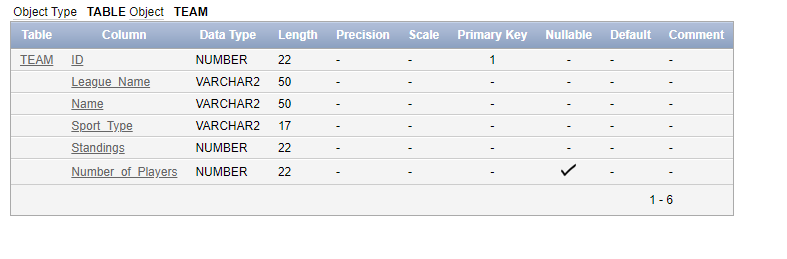
1. . DESC STAFF



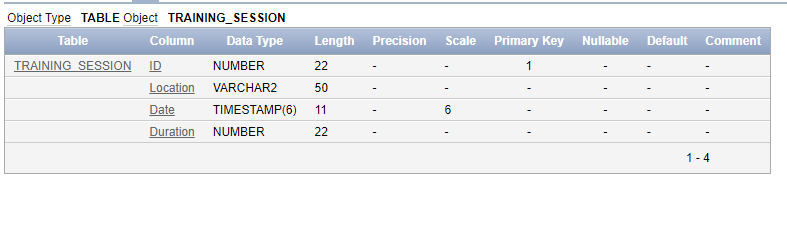
1. DESC VENUE



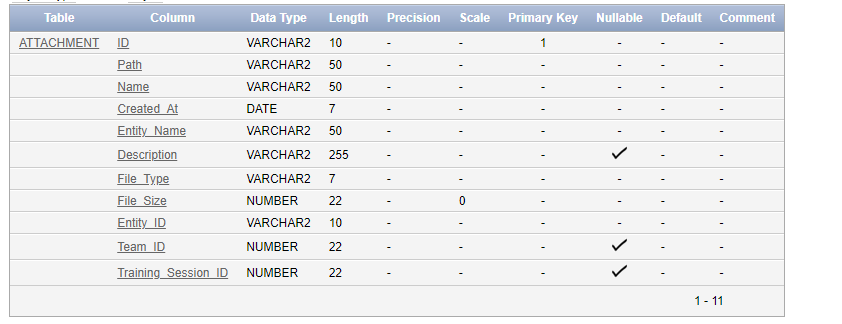
1. DESC TEAM



1. DESC TRAINING\_SESSION



6. DESC TRAINING\_DRILL

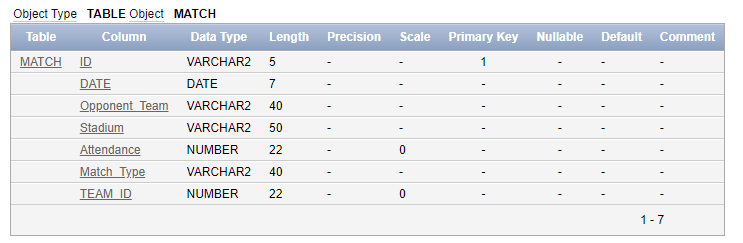


7. DESC TROPHY

A screenshot of a computer

Description automatically generated

8. DESC MATCH



9. DESC CONTRACT

A screenshot of a computer

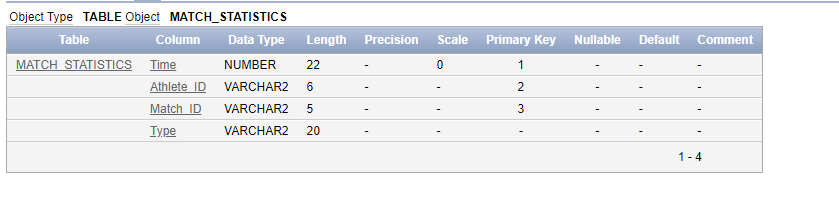
Description automatically generated

10. DESC MEDICAL\_RECORD

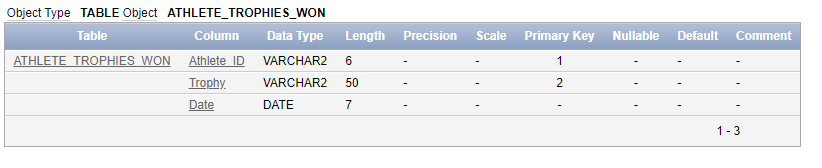
A screenshot of a computer

Description automatically generated

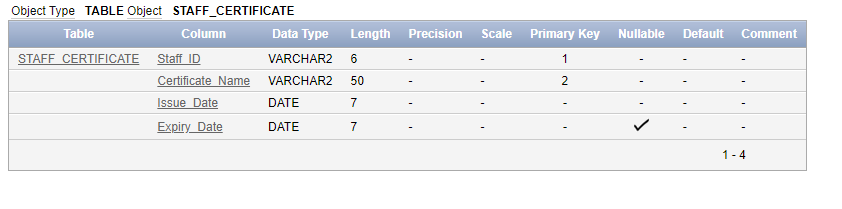
11. DESC MATCH\_STATISTICS



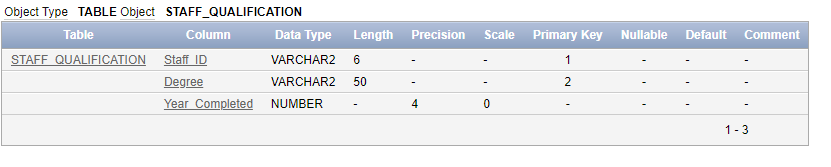
1. DESC ATHLETE\_TROPHIES\_WON



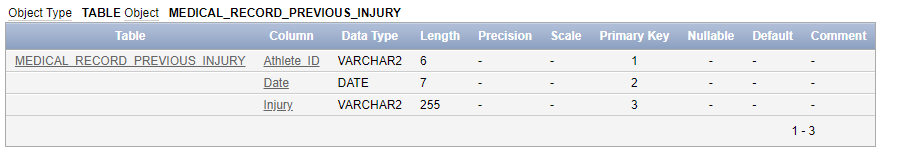
13. DESC STAFF\_CERTIFICATE



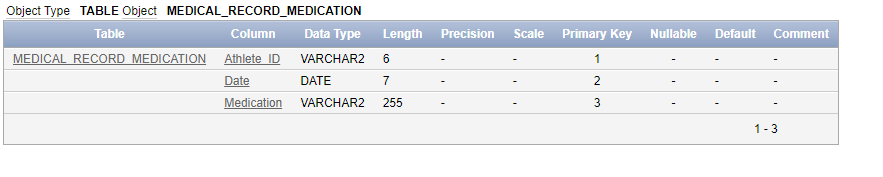
14. DESC STAFF\_QUALIFICATION



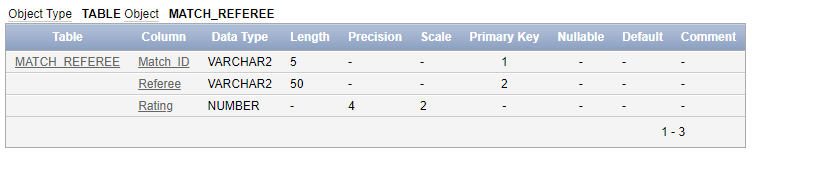
15. DESC MEDICAL\_RECORD\_PREVIOUS\_INJURY



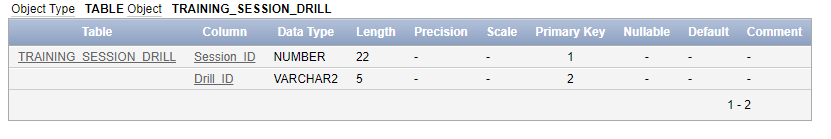
16. DESC MEDICAL\_RECORD\_MEDICATION

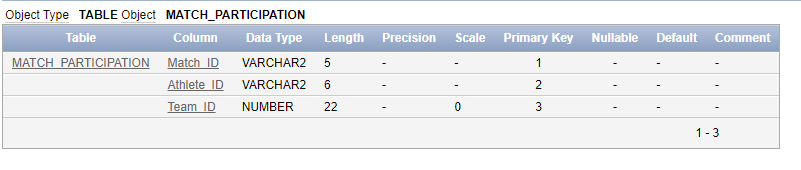


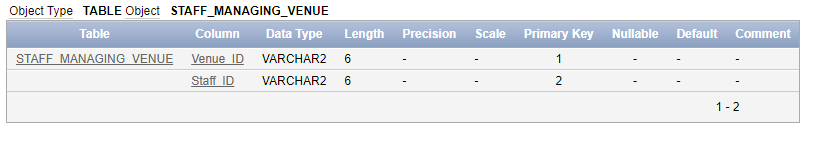
17. DESC MATCH\_REFEREE



18. DESC TRAINING\_SESSION\_DRILL



19. DESC MATCH\_PARTICIPATION

20. STAFF\_MANGING\_VENUE

**IX.Queries**

It is time to perform some transactions on the data to test the sports club system!

**Query 1:** Provide an outstanding salary increase for staff members holding significant positions.

UPDATE CONTRACT SET "Salary" = 10000000 WHERE ID IN (

SELECT "Contract\_ID" FROM Staff WHERE ID IN (

SELECT ID FROM Staff WHERE "Role" = 'Head Coach'

INTERSECT

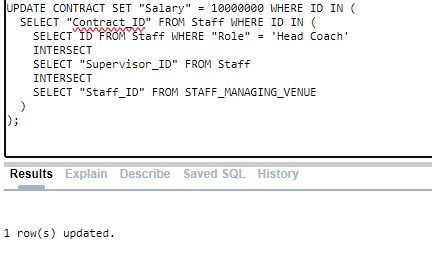
SELECT "Supervisor\_ID" FROM Staff

INTERSECT

SELECT "Staff\_ID" FROM STAFF\_MANAGING\_VENUE

)

);



A screenshot of a computer

Description automatically generated

**Query 2:** Retrieve the team that has scored over 3 points, while the referee's rating is below six.

SELECT \* FROM MATCH WHERE ID IN (

SELECT "Match\_ID" as ID

FROM "MATCH\_STATISTICS"

WHERE "Type" = 'Goal'

GROUP BY "Match\_ID"

HAVING COUNT(\*) >= 3

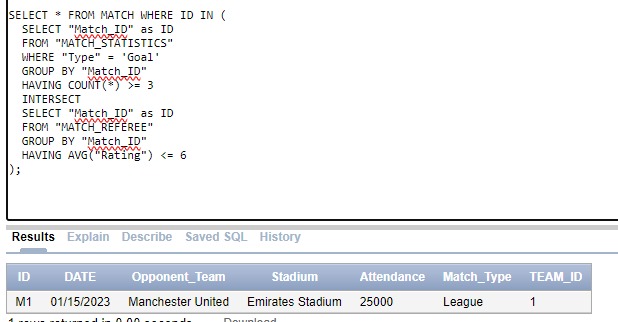
INTERSECT

SELECT "Match\_ID" as ID

FROM "MATCH\_REFEREE"

GROUP BY "Match\_ID"

HAVING AVG("Rating") <= 6)



**Query 3:** In our club, we aim to raise the release clauses, by 65%, for athletes exceeding 180 cm in height and 80 kg in weight, and who have won more than two trophies.

UPDATE "CONTRACT"

SET "Release\_Clause" = "Release\_Clause" \* 1.65

WHERE "ID" IN (

SELECT a."Contract\_ID"

FROM ATHLETE a

JOIN MEDICAL\_RECORD mr ON a.id = mr."Athlete\_ID"

JOIN ATHLETE\_TROPHIES\_WON atw ON a.id = atw."Athlete\_ID"

WHERE mr."Height" > 180 AND mr."Weight" > 80

GROUP BY a."Contract\_ID"

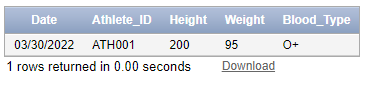
HAVING COUNT(atw."Trophy") > 2

)

Athlete 1 trophies:

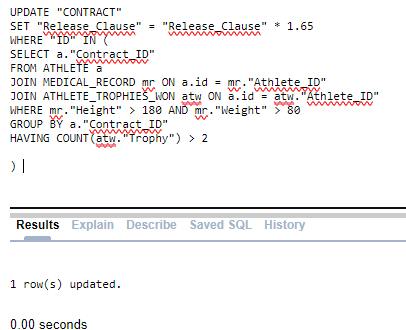


Athlete 1’s Medical Record:



Athlete 1’s Contract status:





Athlete 1’s Contract status:



**Query 4**: Identify the team that has achieved the highest number of trophies and award a bonus of 25% to both the team coach and each athlete.

UPDATE CONTRACT

SET "Bonus" = "Bonus" \* 1.25

WHERE ID IN (

  SELECT "Contract\_ID" AS ID

  FROM STAFF

  WHERE "Team\_ID" IN (

    SELECT Team\_ID

    FROM (

      SELECT t.ID AS Team\_ID, COUNT(tr.ID) AS trophy\_count

      FROM Team t

      JOIN Trophy tr ON t.ID = tr."Team\_ID"

      GROUP BY t.ID

      ORDER BY trophy\_count DESC

    )

  )

  UNION

  SELECT "Contract\_ID" AS ID

  FROM ATHLETE

  WHERE "Team\_ID" IN (

    SELECT Team\_ID

    FROM (

      SELECT t.ID AS Team\_ID, COUNT(tr.ID) AS trophy\_count

      FROM Team t

      JOIN Trophy tr ON t.ID = tr."Team\_ID"

      GROUP BY t.ID

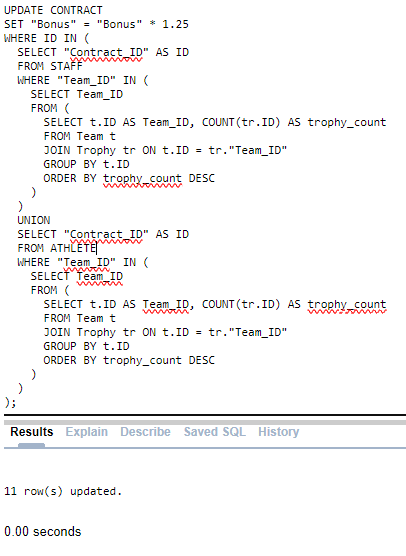
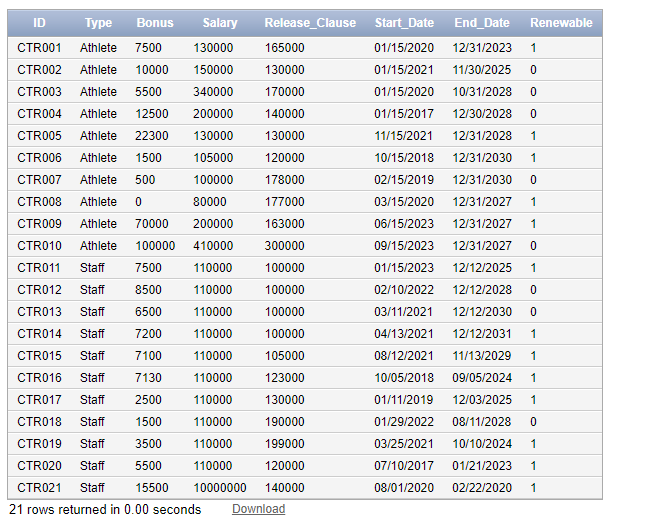
      ORDER BY trophy\_count DESC

    )

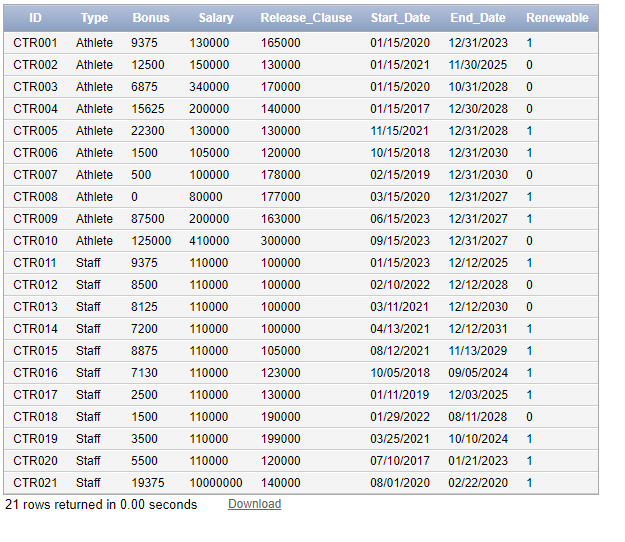
  )

);

Contract Table status before executing the query:



Contract Table status after executing the query:



**Query 5:** Get the match id, athlete first and last name where athlete has more than 3 fouls in the same match.

SELECT M.ID,A."First\_Name",A."Last\_Name"

FROM "MATCH" M

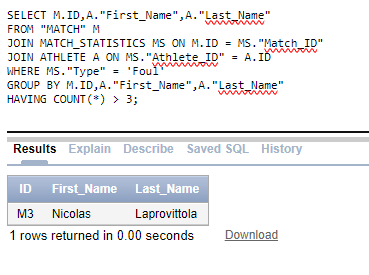
JOIN MATCH\_STATISTICS MS ON M.ID = MS."Match\_ID"

JOIN ATHLETE A ON MS."Athlete\_ID" = A.ID

WHERE MS."Type" = 'Foul'

GROUP BY M.ID,A."First\_Name",A."Last\_Name"

HAVING COUNT(\*) > 3;



**Query 6:** Retrieve Athlete (ID, FirstName, LastName and position) and match\_id when the athlete has committed a foul but didn’t receive a card.

SELECT DISTINCT A."ID",A."First\_Name",A."Last\_Name",A."Position",M.ID

FROM ATHLETE A

JOIN MATCH\_STATISTICS MS ON A.ID = MS."Athlete\_ID"

JOIN "MATCH" M ON MS."Match\_ID" = M.ID

WHERE MS."Type" = 'Foul'

  AND M."Match\_Type" = 'League'

  AND M."TEAM\_ID" IN (SELECT ID FROM TEAM WHERE "Sport\_Type"= 'Football')

  AND A.ID NOT IN (

    SELECT MSA."Athlete\_ID"

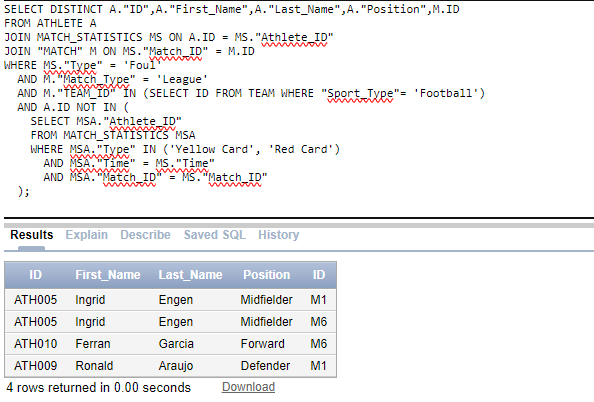
    FROM MATCH\_STATISTICS MSA

    WHERE MSA."Type" IN ('Yellow Card', 'Red Card')

      AND MSA."Time" = MS."Time"

      AND MSA."Match\_ID" = MS."Match\_ID"

  );



A table with numbers and letters

Description automatically generated

**Query 7**: we need to find a venue that can hold more than 500 people with the type of Banquet Hall

SELECT "VENUE"."Name", "VENUE"."Capacity", “VENUE”.”Type", VENUE"."Phone\_Number", "STAFF"."Email" AS "Employee\_Email"

FROM "VENUE"

JOIN "MANAGES" ON "VENUE"."ID" = "MANAGES"."Venue\_ID"

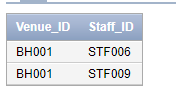
JOIN "STAFF" ON "MANAGES"."Staff\_ID" = "STAFF"."ID"

WHERE "VENUE"."Capacity" >= 500 AND “VENUE”.”Type” = “Banquet Hall”;

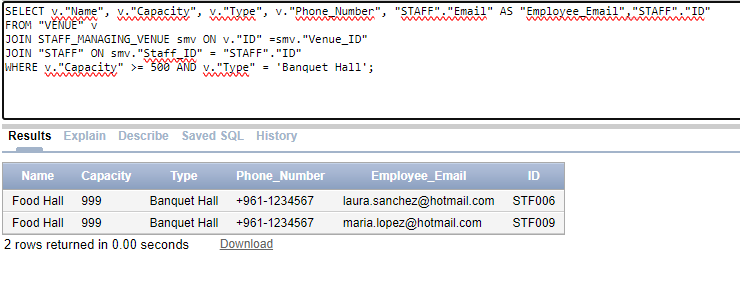
VENUE BH001



STAFF managing BH001:



OUPUT OF QUERY:



**Query 8:** retrieve medical employees whose medical certificates have expired to maintain our clubs integrity

SELECT \*

FROM "STAFF"

WHERE "Role" = 'Medical Team'

AND "ID" IN (

    SELECT "Staff\_ID"

    FROM "STAFF\_CERTIFICATE"

    WHERE "Expiry\_Date" < SYSDATE

);



**Query 9:** retrieve all staff that have a higher salary than their supervisors.

SELECT stf1."First\_Name"||' ' ||  stf1."Last\_Name" as Supervisee\_Name, stf2."First\_Name"|| '   ' ||  stf2."Last\_Name" as Supervisor\_Name

FROM STAFF stf1

JOIN STAFF stf2 ON stf1."Supervisor\_ID" = stf2.ID

WHERE stf2.ID IS NOT NULL

  AND (SELECT "Salary" FROM CONTRACT WHERE id = stf1."Contract\_ID") > (SELECT "Salary" FROM CONTRACT WHERE id = stf2."Contract\_ID");

STAFF:



CONTRACT:

A screenshot of a computer

Description automatically generated

OUTPUT:

A screenshot of a computer

Description automatically generated

**Query 10:** we want to check if an athlete had an injury a month prior to the game to let them skip the match.

UPDATE "MATCH\_PARTICIPATION"

SET "Match\_ID" = NULL

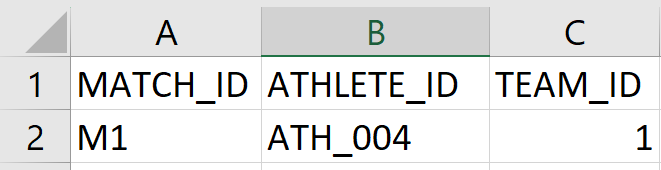
WHERE "Athlete\_ID" IN (

    SELECT "Athlete\_ID" , “DATE”

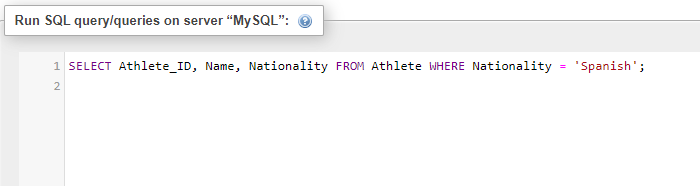
    FROM "MEDICAL\_RECORD\_PREVIOUS\_INJURY" , “MATCH”

    WHERE "Date" BETWEEN “MATCH”.”Date” - 30 AND “

);



**Query 11**: query to find all athletes who has the Spanish nationality.

****

**OUTPUT:**

**A white background with black text

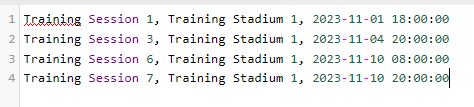
Description automatically generated**

**Qurey12:** query to list all training sessions scheduled at a specific venue.

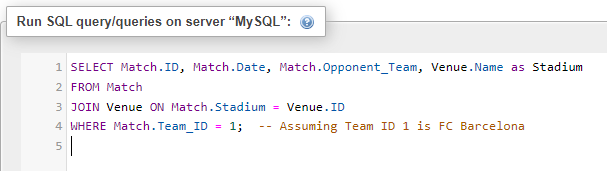
**A screenshot of a computer

Description automatically generated**

**OUTPUT:**

****

**Qurey13:** retrieve all matches with their locations and dates for a specific team id which is (FC Barcelona) since team fc Barcelona has a team id of 1.

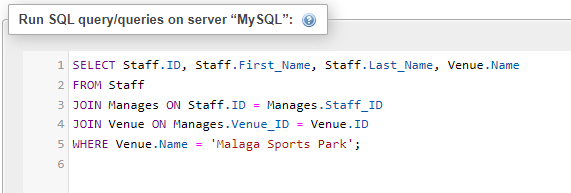
****

**OUTPUT:**

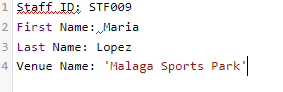
**A close-up of a screen

Description automatically generated**

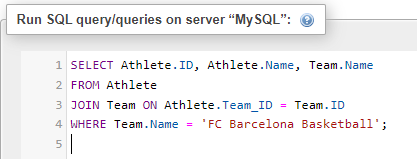
**Qurey14:** query that lists all staff members managing a specific venue, 'Malaga Sports Park'.

****

**OUTPUT:**

****

**Qurey15**: find all athletes from a specific team which is FC Barcelona Basketball.

****

**OUTPUT:**

**A group of black text

Description automatically generated**

**Qurey16:** query to list all training drills by difficulty level.

**A screenshot of a computer program

Description automatically generated**

**OUTPUT:**

**A black text on a white background

Description automatically generated**

**Qurey17:** query to show contract details for staff with a specific role:

**A screenshot of a computer

Description automatically generated**

**OUTPUT :**

**A group of black text

Description automatically generated**

**Qurey18:** query to find all matches played by a specific team:

**A screenshot of a computer

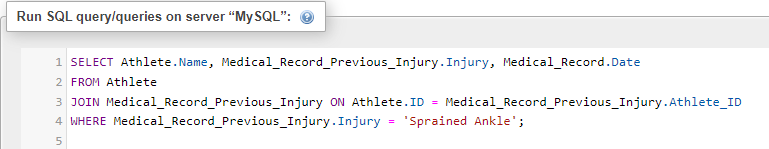
Description automatically generated**

**Output:**

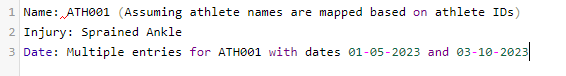
**A close-up of a number

Description automatically generated**

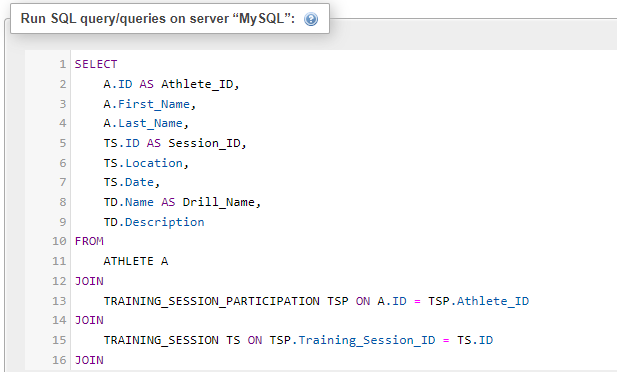
**Qurey19:** retrieve medical records for athletes with a specific injury ('Sprained Ankle'):

****

**OUTPUT:**

****

**Qurey20:** query to list all athletes and their training sessions along with drills:

**** **A computer code with text

Description automatically generated**

OUTPUT:

**A screenshot of a computer

Description automatically generated**

**X Conclusion**

This report detailed the development of a database application system for a sports club. The initial phase started with the creation of an Entity-Relationship Diagram (ERD), followed by the application of a seven-step mapping algorithm to translate the conceptual design into a structured and logical design. This algorithm played a crucial role in mapping regular entities, weak entities, relationships, and multi-valued attributes, utilizing foreign keys and partial keys.

In this project, the main focus was on the recreation of tables and the insertion of data . This phase involved formulating a minimum of 20 distinct SQL transactions designed to query the database. These queries were then executed against the Oracle database to validate their functionality.

The overall skills gained from this project included proficiency in using SQL to design, implement, and query a relational database system for a sports club.