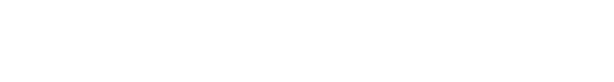
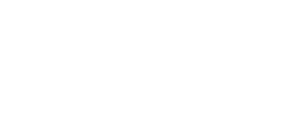
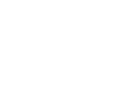
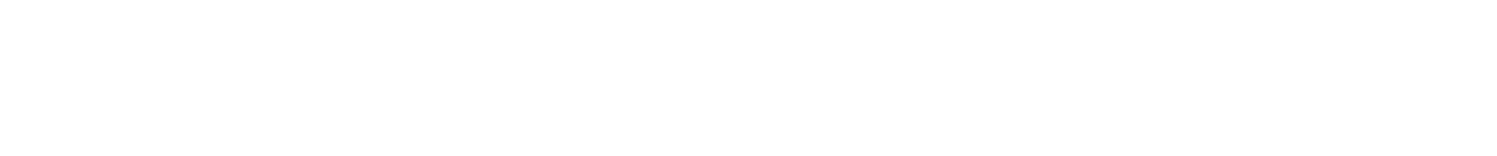
# GROUP ASSIGNMENT COVERSHEET



**KING’S OWN INSTITUTE\***

**Success in Higher Education**

Unit Name: : ICT713 - Advanced Database Design and Development

Lecturer’s name: DR. PRABHU JYOT SINGH \_\_\_\_\_\_\_\_\_

Assignment Title: Database Project – Group Assessment

**Declaration**

(This declaration must be completed by all students in the group or the assignment will not be marked.)

We, the undersigned, certify the following:

* We have read and understood the *Student Academic Misconduct Policy*
* This assignment is our own work based on our personal study and or research.
* We have acknowledged all material and sources used in the preparation of this assignment including any material generated in the course of our employment.
* **The assignment has not previously been submitted for assessment in this or any other unit.**
* We have not copied in part or in whole or otherwise plagiarised the work of other students.
* We have read and understand the criteria used for assessment.
* The assignment is within the word and page limits specified in the unit outline.
* The use of any material in this assignment does not infringe the intellectual property / copyright of a third party.
* We understand that this assignment may undergo electronic detection for plagiarism, and an anonymous copy of the assignment may be retained on the database and used to make comparisons with other assignments in future.
* By completing this coversheet in full and submitting this assignment electronically, we are bound by the conditions of the KOI's *Student Academic Misconduct Policy* and the declaration on this coversheet.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | Family Name | Given Name(s) | Student ID | Tutorial Code | Signature |
| Student 1 | Alonzo | Jerico | 20019490 | Group 2 | Jeo\_A |
| Student 2 | Alvarado | Beatriz | 12301095 | Group 2 | beaa. |
| Student 3 | Austria | John Benny | 12301175 | Group 2 | J\_AUST |
| Student 4 | Patawaran | Jeffrey | 12301039 | Group 2 | JGP |
| Student 5 |  |  |  |  |  |

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

**Assignment Receipt**

Unit Name: ICT713 - Advanced Database Design Lecturer’s Name: DR. PRABHU JYOT SINGH

Assignment Title: Database Project – Group Assessment

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | Family Name | Given Name(s) | Student ID | Tutorial Code | Signature |
| Student 1 | Alonzo | Jerico | 20019490 | Group 2 | Jeo\_A |
| Student 2 | Alvarado | Beatriz | 12301095 | Group 2 | beaa. |
| Student 3 | Austria | John Benny | 12301175 | Group 2 | J\_AUST |
| Student 4 | Patawaran | Jeffrey | 12301039 | Group 2 | JGP |
| Student 5 |  |  |  |  |  |

GROUP ASSIGNMENT COVER SHEET \* AUSTRALIAN INSTITUTE OF BUSINESS AND MANAGEMENT PTY LTD PAGE **1** OF **1**

1. JULY 2015 CRICOS 03171A ABN: 72 132 629 97

# TABLE OF CONTENTS

[GROUP ASSIGNMENT COVERSHEET 1](#_Toc155484823)

[TABLE OF CONTENTS 2](#_Toc155484824)

[LIST OF FIGURES 3](#_Toc155484825)

[LIST OF TABLES 4](#_Toc155484826)

[1.0. INTRODUCTION 5](#_Toc155484827)

[2.0. PROJECT SCENARIO: MUSIC AND ENTERTAINMENT DATABASE 6](#_Toc155484828)

[3.0. BUSINESS RULES 7](#_Toc155484829)

[3.1. LIST OF BUSINESS RULES 7](#_Toc155484830)

[3.2. BUSINESS RULES IN CROW’S FOOT NOTATION 8](#_Toc155484831)

[4.0. ENTITIES WITH THEIR ATTRIBUTES 12](#_Toc155484832)

[5.0. NORMALIZATION 13](#_Toc155484833)

[6.0. ENTITY RELATIONSHIP DIAGRAM 18](#_Toc155484834)

[7.0. EXTENDED ENTITY RELATIONSHIP DIAGRAM 19](#_Toc155484835)

[8.0. CREATION OF TABLES AND DATA POPULATION 21](#_Toc155484836)

[8.1. TABLE CREATION 21](#_Toc155484837)

[8.2. DATA POPULATION 34](#_Toc155484838)

[9.0. SQL QUERIES 45](#_Toc155484839)

[9.1. SELECT query with ORDER BY clause 45](#_Toc155484840)

[9.2. SELECT query with WHERE clause 46](#_Toc155484841)

[9.3. INNER JOIN Query 46](#_Toc155484842)

[9.4. LEFT JOIN Query 47](#_Toc155484843)

[9.5. RIGHT JOIN Query 48](#_Toc155484844)

[9.6. FULL JOIN Query 48](#_Toc155484845)

[10.0 REPORT 50](#_Toc155484846)

[10.1. SELECT query with ORDER BY clause 50](#_Toc155484847)

[10.2. SELECT query with WHERE clause 50](#_Toc155484848)

[10.3. INNER JOIN Query 51](#_Toc155484849)

[10.4. LEFT JOIN Query 52](#_Toc155484850)

[10.5. RIGHT JOIN Query 53](#_Toc155484851)

[10.6. FULL JOIN Query 54](#_Toc155484852)

[11.0 REFERENCES 55](#_Toc155484853)

# LIST OF FIGURES

[Figure 3‑1. PERFORMER-SETLIST CFN 8](#_Toc155484564)

[Figure 3‑2 ROAD\_MANAGER-PERFORMER CFN 8](#_Toc155484565)

[Figure 3‑3 EXECUTIVE-ROAD\_MANAGER CFN 9](#_Toc155484566)

[Figure 3‑4 EXECUTIVE-DEPARTMENT CFN 9](#_Toc155484567)

[Figure 3‑5 DEPARTMENT-STAFF CFN 10](#_Toc155484568)

[Figure 3‑6 PERFORMER-CONCERT CFN 10](#_Toc155484569)

[Figure 3‑7 CONCERT-STADIUM CFN 11](#_Toc155484570)

[Figure 3‑8 STADIUM-TICKET CFN 11](#_Toc155484571)

[Figure 3‑9 TICKET-ATTENDEE CFN 12](#_Toc155484572)

[Figure 5‑1 EXECUTIVE (3NF) 14](#_Toc155484573)

[Figure 5‑2 DEPARTMENT (3NF) 14](#_Toc155484574)

[Figure 5‑3 STAFF (3NF) 15](#_Toc155484575)

[Figure 5‑4 ROAD\_MANAGER (3NF) 15](#_Toc155484576)

[Figure 5‑5 PERFORMER (3NF) 15](#_Toc155484577)

[Figure 5‑6 SETLIST (3NF) 15](#_Toc155484578)

[Figure 5‑7 CONCERT (3NF) 16](#_Toc155484579)

[Figure 5‑8 TICKET (3NF) 16](#_Toc155484580)

[Figure 5‑9 STADIUM (3NF) 16](#_Toc155484581)

[Figure 5‑10 ATTENDEE (3NF) 16](#_Toc155484582)

[Figure 5‑11 SECURITY (3NF) 17](#_Toc155484583)

[Figure 5‑12 TECHNICAL (3NF) 17](#_Toc155484584)

[Figure 5‑13 EVENT\_CREW (3NF) 17](#_Toc155484585)

[Figure 6‑1 BLM ERD 18](#_Toc155484586)

[Figure 7‑1 BLM EERD 19](#_Toc155484587)

[Figure 8‑1 Executive Table 21](#_Toc155484588)

[Figure 8‑2 Department Table 22](#_Toc155484589)

[Figure 8‑3 Staff Table 23](#_Toc155484590)

[Figure 8‑4 Road Manager Table 24](#_Toc155484591)

[Figure 8‑5 Performer Table 25](#_Toc155484592)

[Figure 8‑6 Setlist Table 26](#_Toc155484593)

[Figure 8‑7 Stadium Table 27](#_Toc155484594)

[Figure 8‑8 Concert Table 28](#_Toc155484595)

[Figure 8‑9 Attendee Table 29](#_Toc155484596)

[Figure 8‑10 Ticket Table 30](#_Toc155484597)

[Figure 8‑11 Security Table 31](#_Toc155484598)

[Figure 8‑12 Technical Table 32](#_Toc155484599)

[Figure 8‑13 Event\_Crew Table 33](#_Toc155484600)

[Figure 8‑14 Executive Records 34](#_Toc155484601)

[Figure 8‑15 Department Records 34](#_Toc155484602)

[Figure 8‑16 Road Manager Records 35](#_Toc155484603)

[Figure 8‑17 Performer Records 36](#_Toc155484604)

[Figure 8‑18 Setlist Records 36](#_Toc155484605)

[Figure 8‑19 Stadium Records 37](#_Toc155484606)

[Figure 8‑20 Attendee Records 38](#_Toc155484607)

[Figure 8‑21 Ticket Records 40](#_Toc155484608)

[Figure 8‑22 Concert Records 40](#_Toc155484609)

[Figure 8‑23 Staff Records 42](#_Toc155484610)

[Figure 8‑24 Security Records 43](#_Toc155484611)

[Figure 8‑25 Technical Records 43](#_Toc155484612)

[Figure 8‑26 Event\_Crew Records 44](#_Toc155484613)

[Figure 9‑1 Query1: Select-Order by 45](#_Toc155484614)

[Figure 9‑2 Query2: Select-Where 46](#_Toc155484615)

[Figure 9‑3 Query3: Inner Join 46](#_Toc155484616)

[Figure 9‑4 Query4: Left Join 47](#_Toc155484617)

[Figure 9‑5 Query5: Right Join 48](#_Toc155484618)

[Figure 9‑6 Query6: Full Join 49](#_Toc155484619)

[Figure 10‑1 Report1 50](#_Toc155484620)

[Figure 10‑2 Report2 50](#_Toc155484621)

[Figure 10‑3 Report3 51](#_Toc155484622)

[Figure 10‑4 Report4 52](#_Toc155484623)

[Figure 10‑5 Report5 53](#_Toc155484624)

[Figure 10‑6 Report6 54](#_Toc155484625)

# LIST OF TABLES

[Table 5‑1 Executive Table 14](#_Toc155484662)

# INTRODUCTION

Over the past two decades, there have been significant developments in the entertainment industry. Technological advancements have increased accessibility to and availability of various forms of entertainment for people worldwide. Database technology is one field that has had the most impact on the entertainment sector.

For businesses that adopt a data-driven culture, the vast amount of data produced by this enhanced connection may lead to revenue growth despite its challenges. Adopting a data-driven approach is the means to boost investment and decision-making boldness in large-scale events and music festivals. Every facet of the entertainment business has been impacted by database technology, including customer service, sales, marketing, predictive tracking, video streaming, and managing taxes, sales, and piracy concerns. Database technology is used by all of the leading entertainment industry organizations.

This report will look into the Bruno Mars Live 2022 concert scenario and then identify the business rules and entities. After placing the business rules and entities, an Entity-Relationship Diagram establishes the relationship between the entities identified in the entertainment. Tables and the population of data are created through MS Access. Finally, queries are made in MS Access to utilise the database in which reports are created entirely.

# PROJECT SCENARIO: MUSIC AND ENTERTAINMENT DATABASE

**"Bruno Mars Live 2022"**

Australia saw a magnificent event in 2022 that drew music enthusiasts from all over the country. A 5-day concert that is celebrated across Australia on Friday, October 14, and Saturday, October 15 in Sydney, Friday, October 21, and Saturday, October 22 in Melbourne and Saturday, October 28 in Brisbane with worldwide sensation Bruno Mars Live and Exclusive. Thousands of people were expected to experience a little "24K Magic" during the whole tour.

The world-famous Bruno Mars agreed to perform at the reopening concerts, bringing his unique style and charisma. Celebrated by Lewis Capaldi as special guest and Nigerian DJ and hit producer SPINALL was announced as a guest act to support Bruno Mars. The Road Manager was essential in arranging all of the artists and working with executive teams to ensure a seamless event. Bruno Mars's carefully chosen setlist promises an unforgettable evening full of his hits and unexpected surprises.

The Executive Team put forth a lot of effort to make sure the event was a success, including a committed Tour Director, Head Producer, Executive Producer, Event Coordinator, and Marketing Director. Their intention was to create a memorable evening that would include music, excitement, and "24K Magic." The event crew members, security, and technical were among the staff who diligently prepared for their duties in overseeing the logistics and guaranteeing the participants' safety.

Bruno Mars, his band members, Lewis Capaldi and SPINALL entered the stage on the evenings of October 14, 15, 21, 22 and 28, and the Stadium was filled with music, dancing, and unleashed excitement. Everyone who was a part of this remarkable night in Sydney's music history was left with a lasting memory by the concert, which drew attendees from all around Australia.

*Draws inspiration from a real-world event while incorporating fictional elements to explore specific themes or concepts relevant to the project. Referencing it from: Teg Dainty. (2022).*

# BUSINESS RULES

The incorporation of business rules is crucial for the modelling and design of information systems. In this report, triggers in relational databases are used to apply business rules. The rule articulates an organisation's operational procedure when executing a job. Business rules encapsulate knowledge about the world, whereas a database is a repository of interconnected information (Raipurkar et al., 2012).

The group established relationship participation rules and limitations and constructed an accurate data model using business rules.

## LIST OF BUSINESS RULES

A setlist is going to be performed by one performer.

A performer can perform one or many setlists.

A road manager manages one performer.

One performer is managed by one road manager.

An executive administers only one road manager.

A road manager administered by one executive.

An executive manages one department.

A department is managed by one executive

A department has one or many staff.

A staff member belongs to one department.

A performer performs in one or many concerts.

A concert can have a performance of one performer.

A concert is held in one stadium.

A stadium can hold one or many concerts.

A stadium can sell one or many tickets.

A ticket is sold in one stadium.

A ticket can be bought by one attendee.

An attendee can buy one or many tickets.

## BUSINESS RULES IN CROW’S FOOT NOTATION

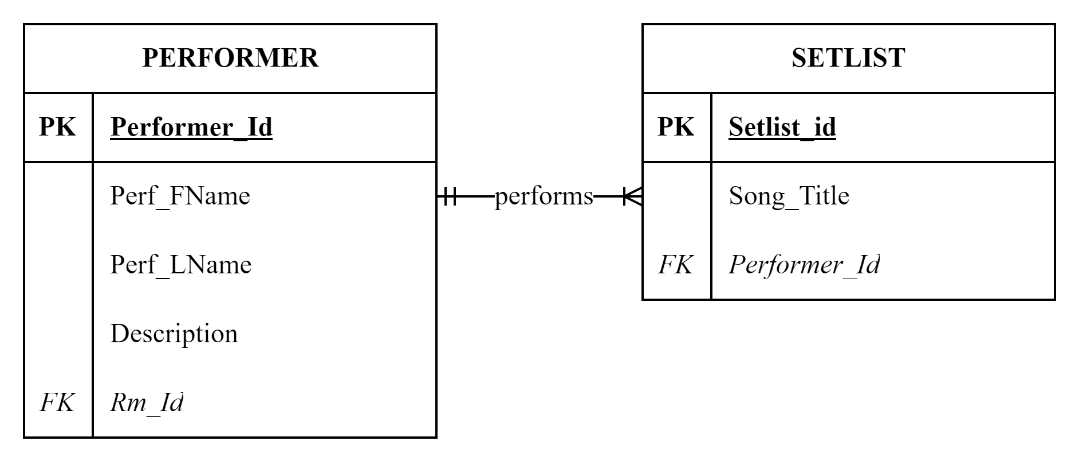
1. PERFORMER and SETLIST 

Figure ‑. PERFORMER-SETLIST CFN

SETLIST to PERFORMER: One-to-One (1:1)

PERFORMER to SETLIST: One-to-Many (1:M)

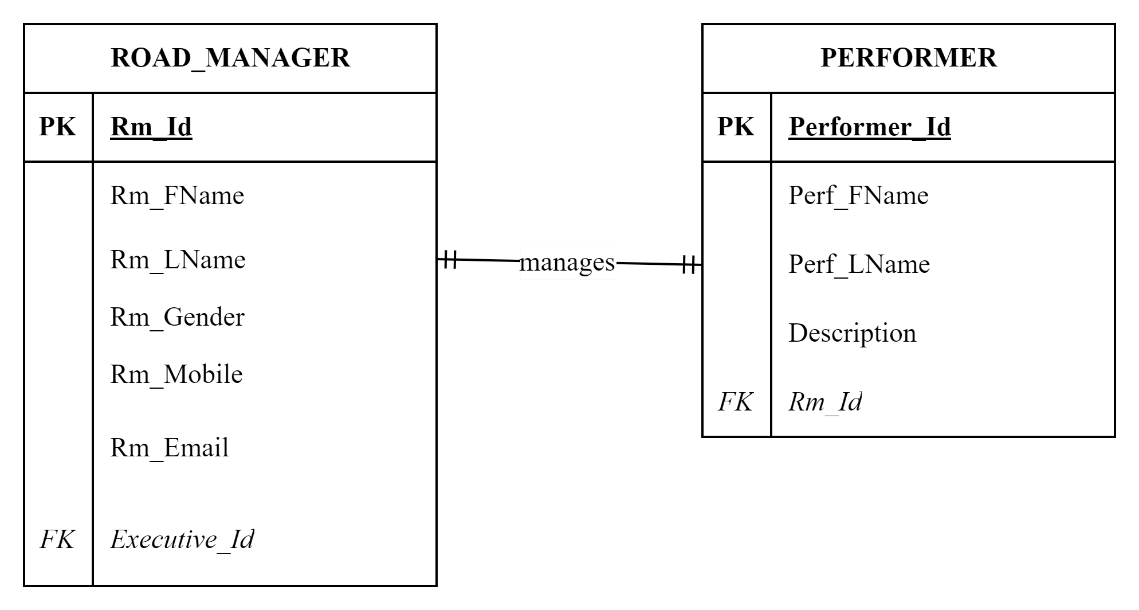
1. ROAD MANAGER and PERFORMER 

Figure ‑ ROAD\_MANAGER-PERFORMER CFN

ROAD MANAGER to PERFORMER: One-to-One (1:1)

PERFORMER to ROAD MANAGER: One-to-One (1:1)

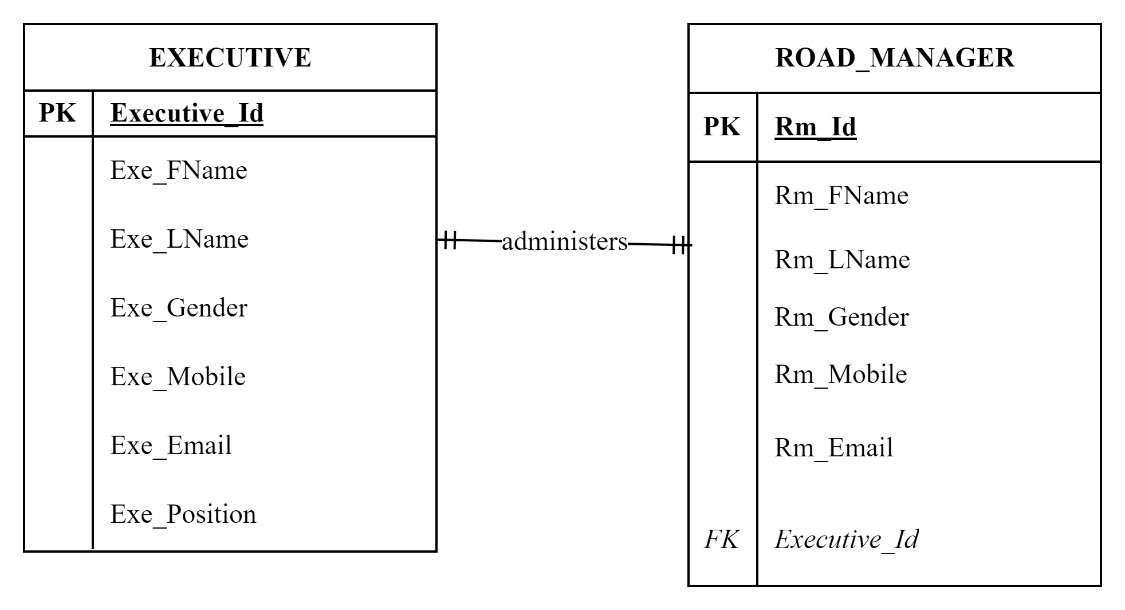
1. EXECUTIVE and ROAD MANAGER 

Figure ‑ EXECUTIVE-ROAD\_MANAGER CFN

EXECUTIVE to ROAD MANAGER: One-to-Many (1:M)

ROAD MANAGER to EXECUTIVE: One-to-One (1:1)

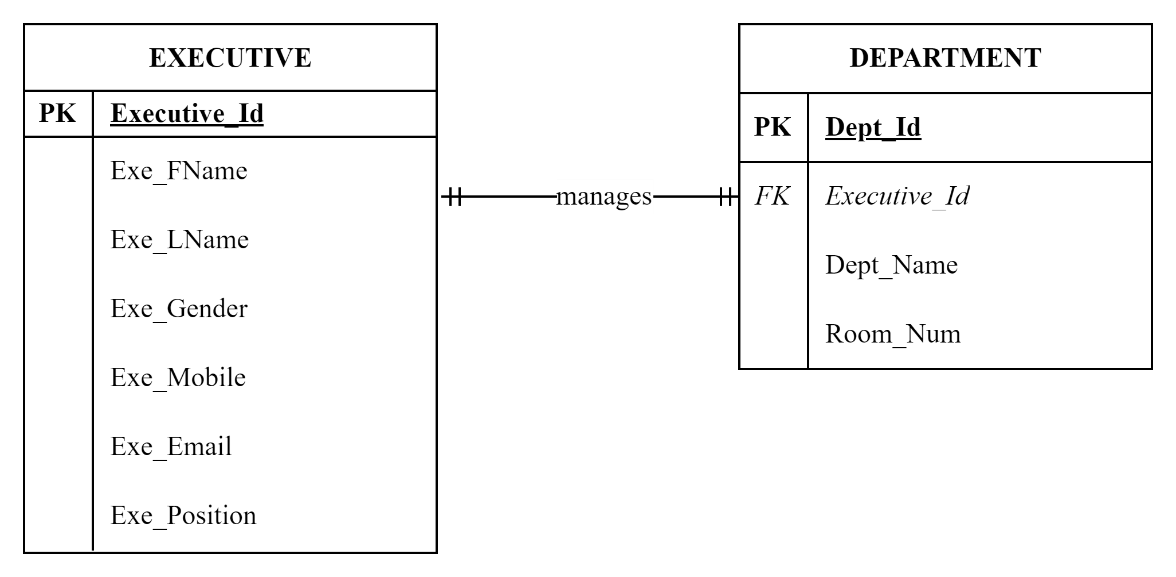
1. EXECUTIVE and DEPARTMENT 

Figure ‑ EXECUTIVE-DEPARTMENT CFN

EXECUTIVE to DEPARTMENT: One-to-One (1:1)

DEPARTMENT to EXECUTIVE: One-to-One (1:1)

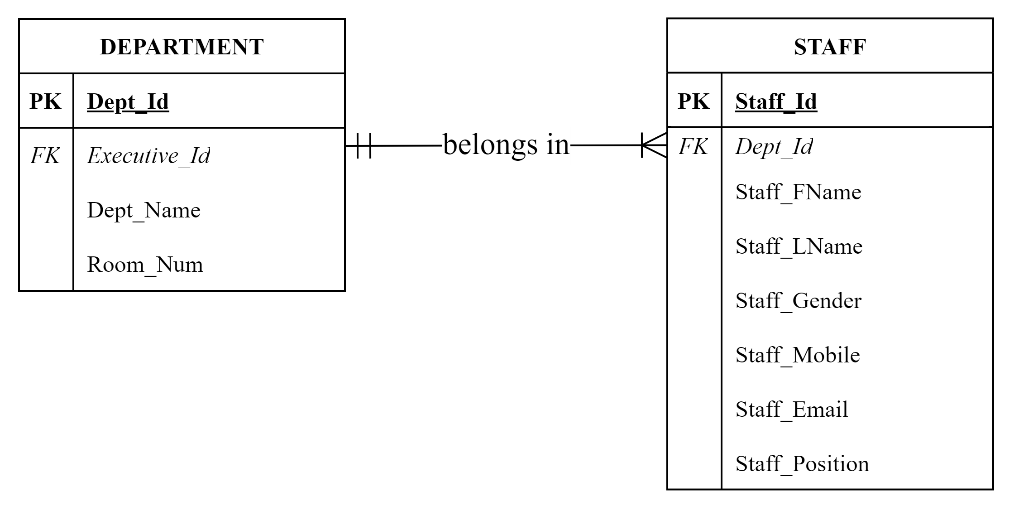
1. DEPARTMENT and STAFF 

Figure ‑ DEPARTMENT-STAFF CFN

DEPARTMENT to STAFF: One-to-Many (1:M)

STAFF to DEPARTMENT: One-to-One (1:1)

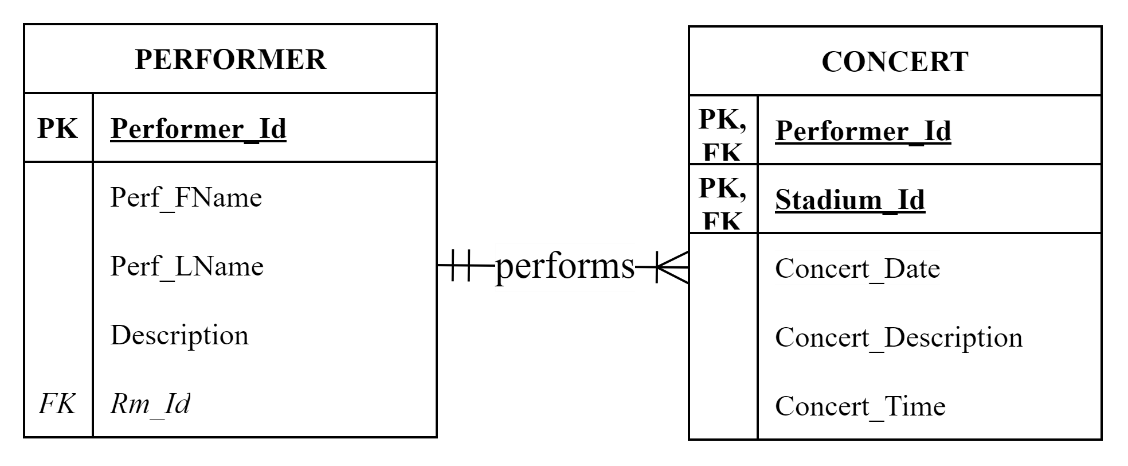
1. PERFORMER and CONCERT 

Figure ‑ PERFORMER-CONCERT CFN

PERFORMER to CONCERT: One-to-Many (1:M)

CONCERT to PERFORMER: One-to-One (1:1)

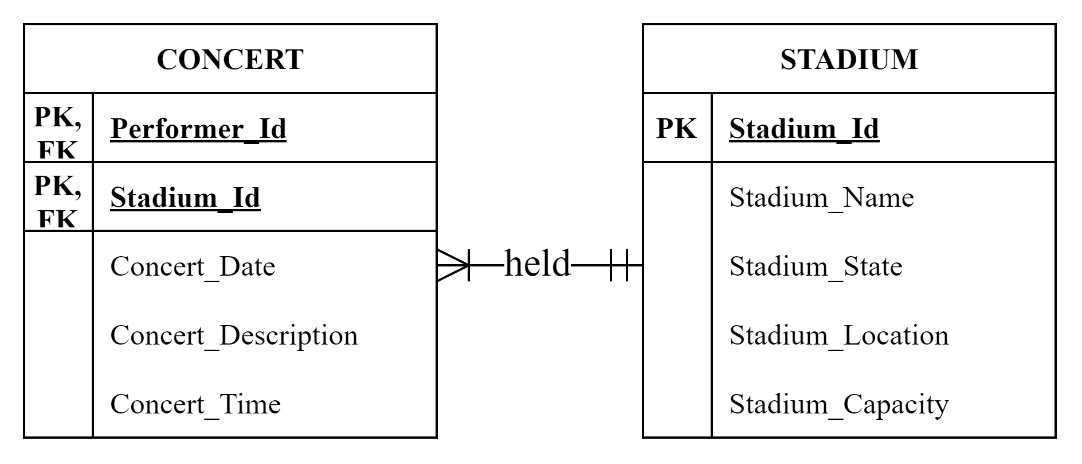
1. CONCERT and STADIUM 

Figure ‑ CONCERT-STADIUM CFN

CONCERT to STADIUM: One-to-One (1:1)

STADIUM to CONCERT: One-to-Many (1:M)

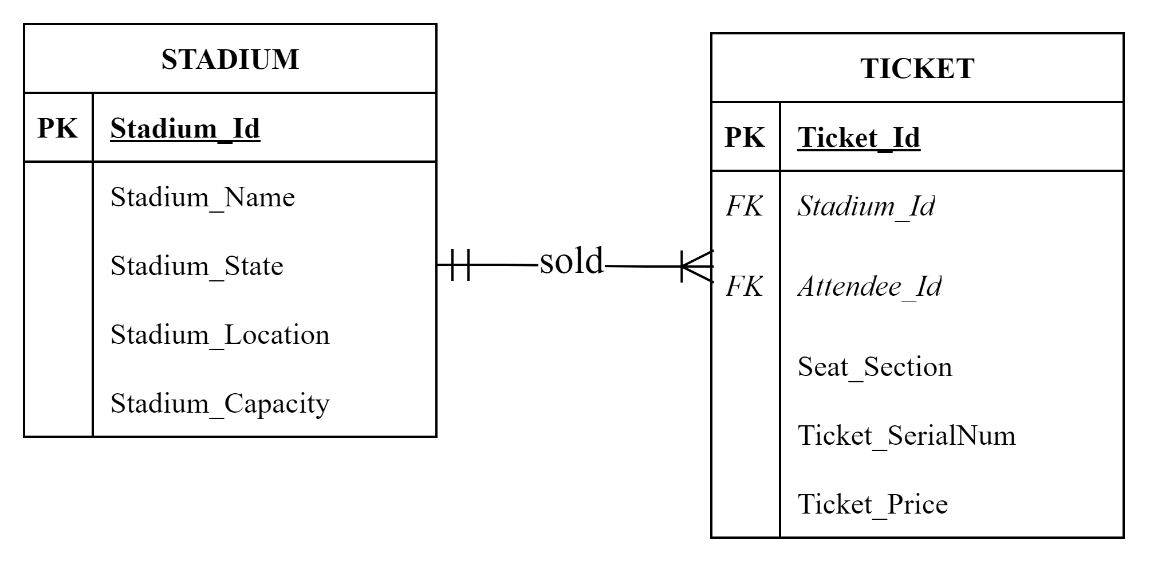
1. STADIUM and TICKET 

Figure ‑ STADIUM-TICKET CFN

STADIUM to TICKET: One-to-Many (1:M)

TICKET to STADIUM: One-to-One (1:1)

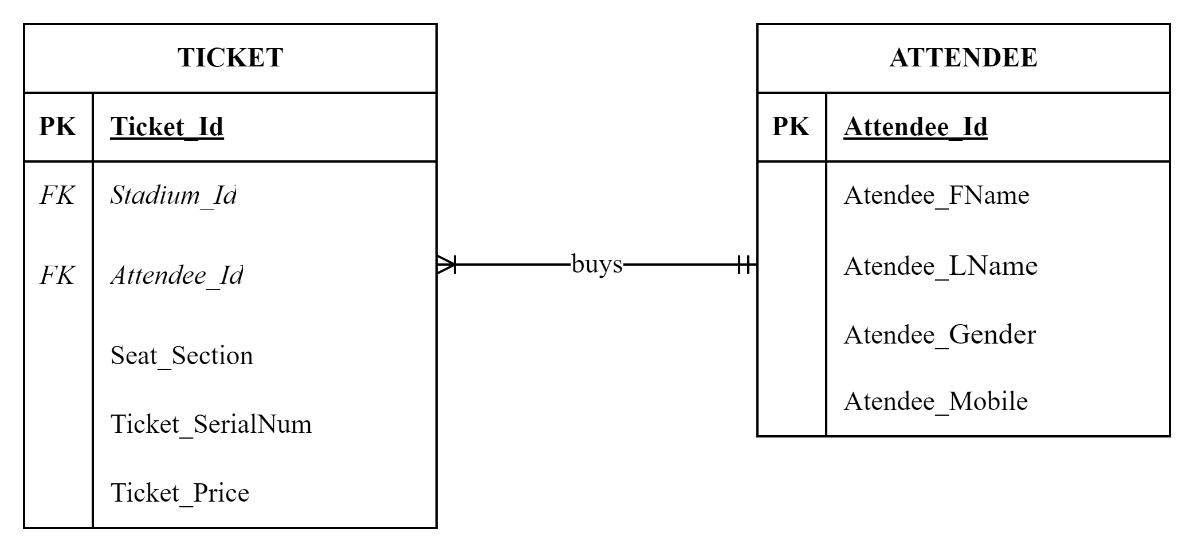
1. TICKET and ATTENDEE 

Figure ‑ TICKET-ATTENDEE CFN

TICKET to ATTENDEE: One-to-One (1:1)

ATTENDEE to TICKET: One-to-Many (1:M)

# ENTITIES WITH THEIR ATTRIBUTES

Entities and their attributes are identified in this chapter. The entities are essential components of databases that store and manage information globally. Entities are valuable for representing and overseeing company data, customer information, and personal records (Halpin et al., 2010). In this music and entertainment database, 13 entities are identified.

**a. EXECUTIVE**

(**Executive\_Id,** Exe\_FName, Exe\_LName, Exe\_Gender, Exe\_Mobile, Exe\_Email, Exe\_Position)

**b. DEPARTMENT**

(**Dept\_Id**, *Executive\_Id* *(f.k.)*, Dept\_Name, Room\_Num)

**c. STAFF**

(**Staff\_Id**, Dept\_Id *(f.k.)*, Staff\_FName, Staff\_LName, Staff\_Gender, Staff\_Mobile, Staff\_Email, Staff\_Position)

**d. ROAD\_MANAGER**

(**Rm\_Id**, Rm\_FName, Rm\_LName, Rm\_Gender, Rm\_Mobile, Rm\_Email, Executive\_Id *(f.k.)*)

**e. PERFORMER**

(**Performer\_Id**, Perf\_Fname, Perf\_LName, Description, Rm\_Id *(f.k.)*)

**f. SETLIST**

(**Setlist\_Id**, Song\_Title, Performer\_Id *(f.k.)*)

**g. CONCERT**

(**Performer\_Id** *(f.k.)***, Stadium\_Id** *(f.k.)***,** Concert\_Date, Concert\_Description, Concert\_Time)

**h. STADIUM**

(**Stadium\_Id,** Stadium\_Name, Stadium\_State, Stadium\_Location, Stadium\_Capacity)

**i. ATTENDEE**

(**Attendee\_Id**, Attendee\_FName, Attendee\_LName, Attendee\_Gender, Attendee\_Mobile)

**j. TICKET**

(**Ticket\_Id**, Stadium\_Id *(f.k.)*, Attendee\_Id *(f.k.)*, Seat\_Section, Ticket\_SerialNum, Ticket\_Price)

**k. SECURITY**

(**Staff\_Id (f.k.),** Security\_License, Police\_Check, Years\_of\_Experience)

**l. TECHNICAL**

(**Staff\_Id (f.k.),** Technical\_Certification, FirstAid\_and\_CPR\_Certification)

**m. EVENT\_CREW**

(**Staff\_Id (f.k.)**, Area\_Designation, Expertise)

# NORMALIZATION

The inherent structure of well-designed databases often aligns with the principles of Third Normal Form (3NF) due to several factors. Modelling real-world entities and relationships, adhering to best practices in normalization, and using common data types like unique identifiers naturally lead to tables that minimize redundancy and dependency (Teorey, 1990).

Table 5-1 depicts the EXECUTIVE table, which is in the 1st Normal Form in DBMS. All characteristics in the table are atomic. Each field name is distinct, and every relation possesses a primary key (Executive\_Id). The domain of attributes remains constant for every value recorded in each table column, as they are of the same type.

Table ‑ Executive Table

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **EXECUTIVE** | | | | | |
| **Executive\_Id** | Exe\_FName | Exe\_LName | Exe\_Gender | Exe\_Mobile | Exe\_Email |
| BMLEX101 | ANDREA | MILLER | F | 425465485 | AndreaMiller@gmail.com |
| BMLEX102 | BRYAN | COOPER | M | 412658794 | BryanCooper@yahoo.com |
| BMLEX103 | STEPHANIE | JONES | F | 415369874 | StephanieJones@gmail.com |
| BMLEX104 | ANDREW | MARTINES | M | 402156486 | AndrewMarties@gmail.com |
| BMLEX105 | BARONE | LEE | M | 420156975 | LeeBarone@Hotmail.com |

The foundation of the second Normal Form (2NF) is complete functional dependency. When a relationship has a primary key that consists of two or more attributes, it is said to have a composite key and is subject to the Second Normal Form (Rolik et al., 2021). To satisfy the requirements of the second standard form, a relation must adhere to the criteria of the first normal form and must not exhibit any instances of partial dependencies. Table 5-1 do not exhibit any partial dependencies and is already in the first normal form (1NF).

All tables in this database adhere to the third standard form (3NF), as they do not exhibit any transitive relationships. Every non-key characteristic is entirely functionally dependent solely on the primary key. The foreign keys (f.k.) establish the relationships between tables.

**a. EXECUTIVE**

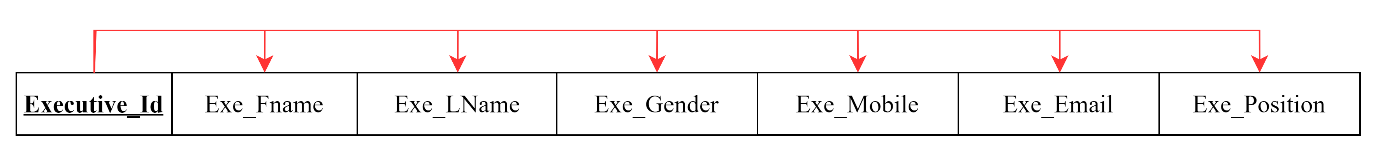


Figure ‑ EXECUTIVE (3NF)

**FD1: Executive\_Id 🡪** Exe\_FName, Exe\_LName, Exe\_Gender, Exe\_Mobile, Exe\_Email, Exe\_Position [3NF, Full Dependency]

**b. DEPARTMENT**

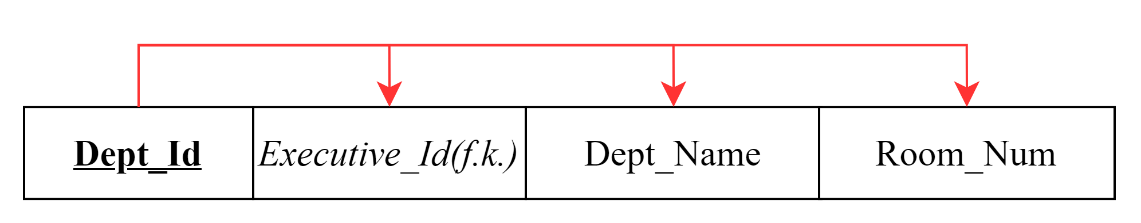


Figure ‑ DEPARTMENT (3NF)

**FD1: Dept\_Id** **🡪** *Executive\_Id* *(f.k.)*, Dept\_Name, Room\_Num [3NF, Full Dependency]

**c. STAFF**

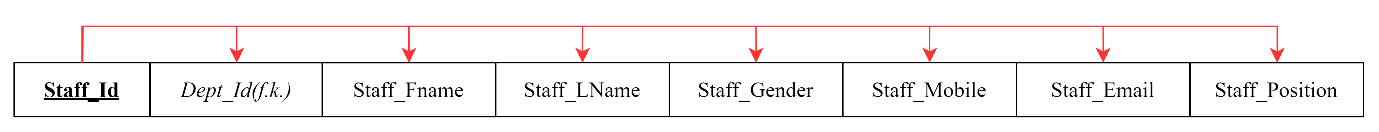


Figure ‑ STAFF (3NF)

**FD1: Staff\_Id** **🡪** *Dept\_Id* *(f.k.)*, Staff\_FName, Staff\_LName, Staff\_Gender, Staff\_Mobile, Staff\_Email, Staff\_Position [3NF, Full Dependency]

**d. ROAD\_MANAGER**

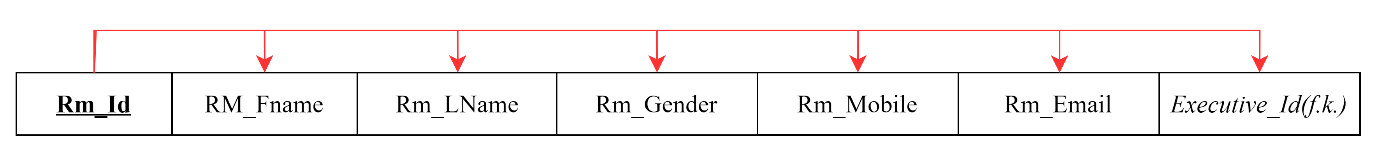


Figure ‑ ROAD\_MANAGER (3NF)

**FD1: Rm\_Id 🡪** Rm\_FName, Rm\_LName, Rm\_Gender, Rm\_Mobile, Rm\_Email, *Executive\_Id* *(f.k.)* [3NF, Full Dependency]

**e. PERFORMER**

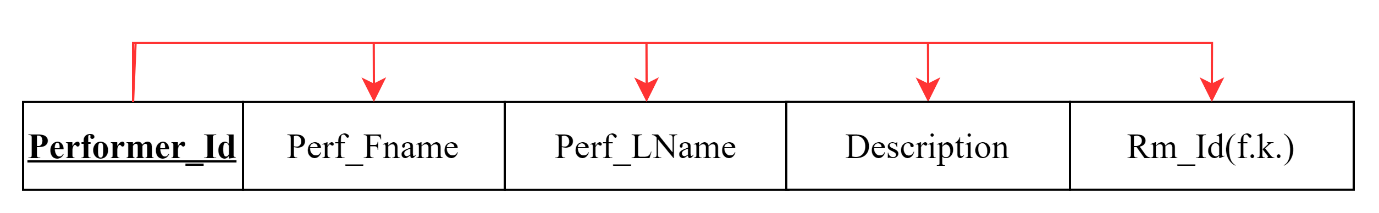


Figure ‑ PERFORMER (3NF)

**FD1: Performer\_Id** **🡪** Perf\_Fname, Perf\_LName, Description, *Rm\_Id* *(f.k.)* [3NF, Full Dependency]

**f. SETLIST**

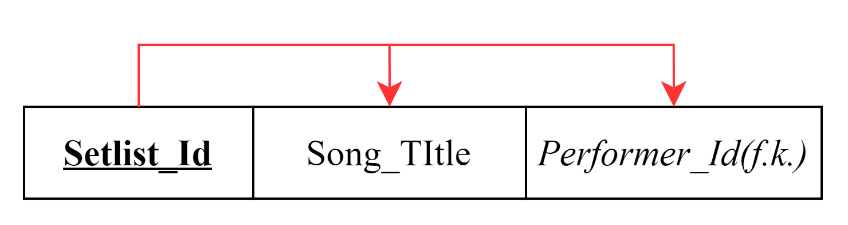


Figure ‑ SETLIST (3NF)

**FD1: Setlist\_Id** **🡪** Song\_Title, *Performer\_Id* *(f.k.)* [3NF, Full Dependency]

**g. CONCERT**

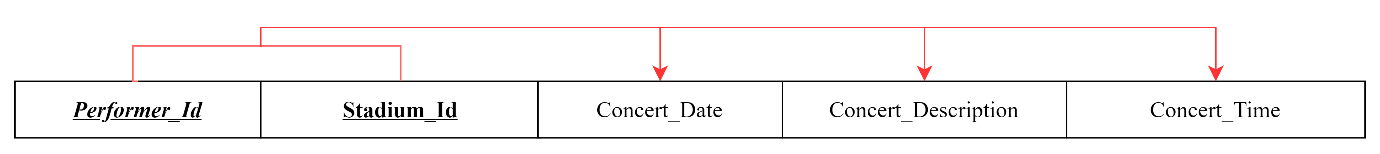


Figure ‑ CONCERT (3NF)

**FD1: Performer\_Id** *(f.k.)***, Stadium\_Id** *(f.k.)* **🡪** Concert\_Date, Concert\_Description, Concert\_Time [3NF, Full Dependency]

**h. TICKET**

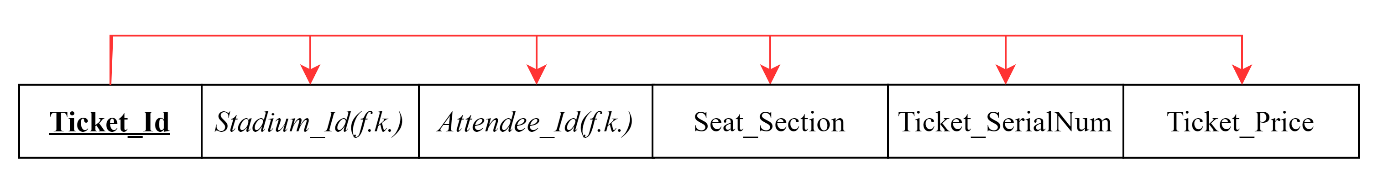


Figure ‑ TICKET (3NF)

**FD1: Ticket\_Id** **🡪** Stadium\_Id *(f.k.)*, Attendee\_Id *(f.k.)*, Seat\_Section, Ticket\_SerialNum, Ticket\_Price [3NF, Full Dependency]

**i. STADIUM**

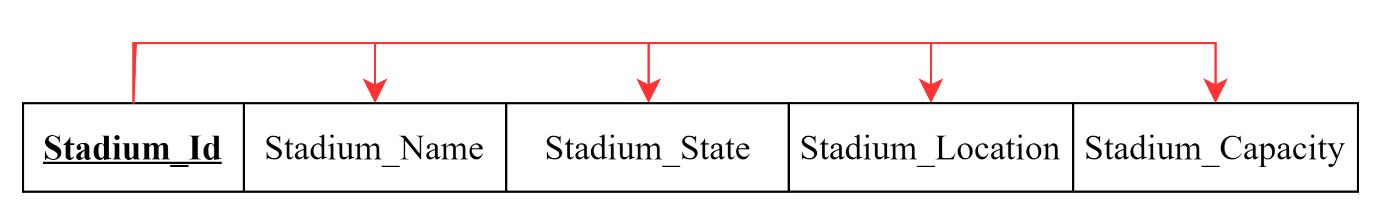


Figure ‑ STADIUM (3NF)

**FD1: STADIUM 🡪 Stadium\_Id,** Stadium\_Name, Stadium\_State, Stadium\_Location, Stadium\_Capacity [3NF, Full Dependency]

**j. ATTENDEE**

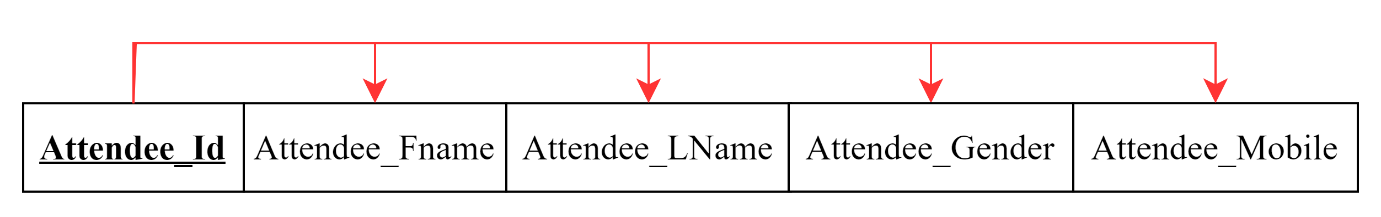


Figure ‑ ATTENDEE (3NF)

**FD1: Attendee\_Id** **🡪** Attendee\_FName, Attendee\_LName, Attendee\_Gender, Attendee\_Mobile [3NF, Full Dependency]

**k. SECURITY**

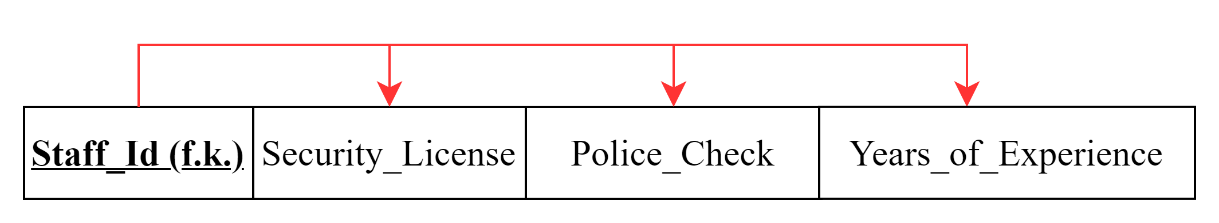


Figure ‑ SECURITY (3NF)

**FD1: Staff\_Id (f.k.) 🡪** Security\_License, Police\_Check, Years\_of\_Experience [3NF, Full Dependency]

**l. TECHNICAL**



Figure ‑ TECHNICAL (3NF)

**FD1: Staff\_Id (f.k.) 🡪** Technical\_Certification, FirstAid\_and\_CPR\_Certification [3NF, Full Dependency]

**m. EVENT\_CREW**

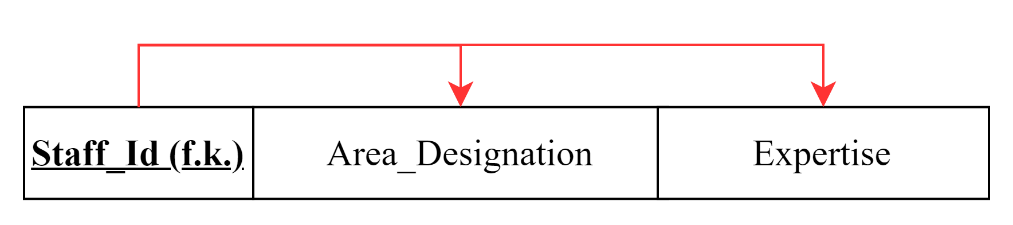


Figure ‑ EVENT\_CREW (3NF)

**FD1: Staff\_Id (f.k.) 🡪** Area\_Designation, Expertise [3NF, Full Dependency]

# ENTITY RELATIONSHIP DIAGRAM

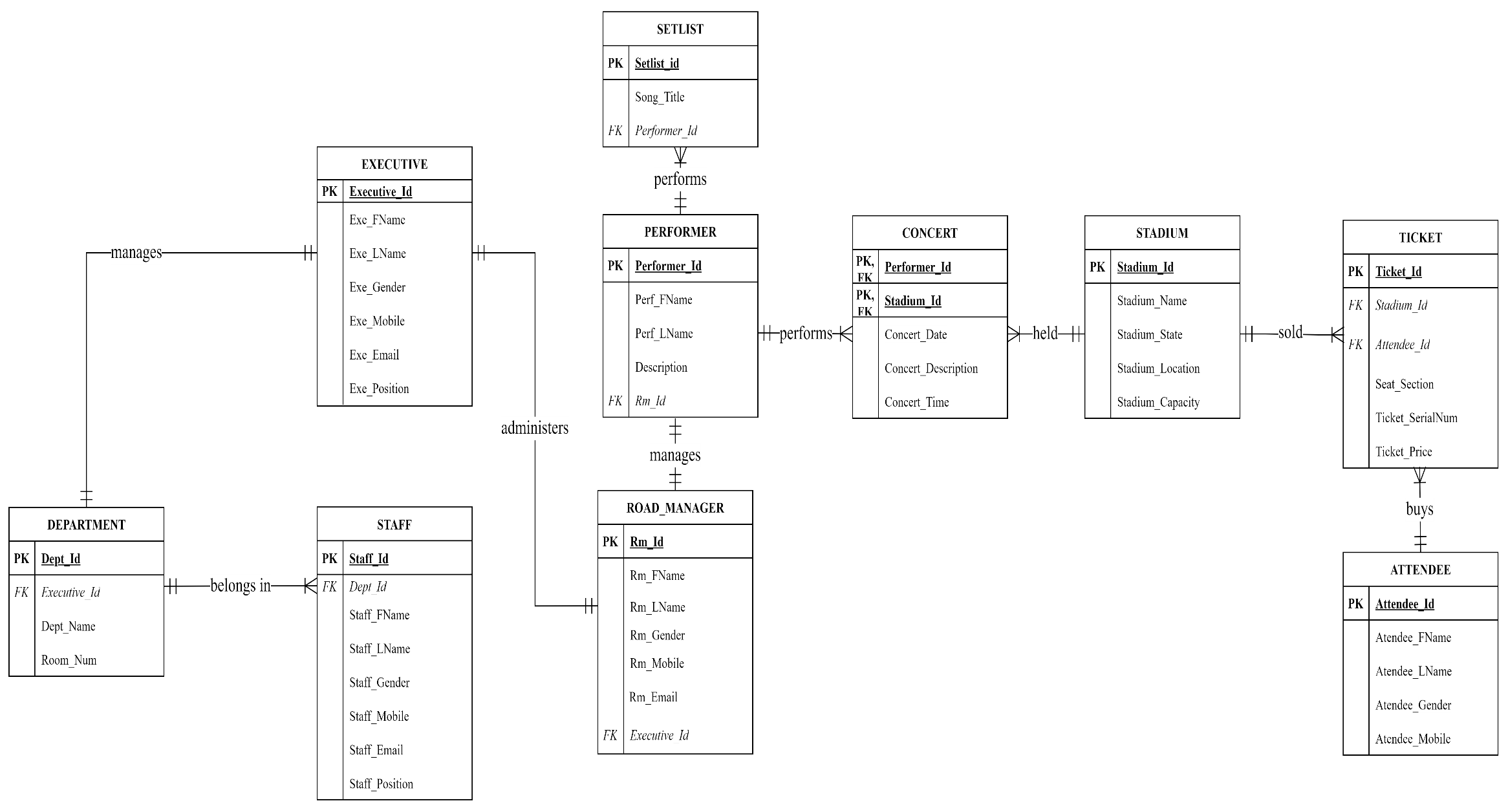


Figure ‑ BLM ERD

# EXTENDED ENTITY RELATIONSHIP DIAGRAM

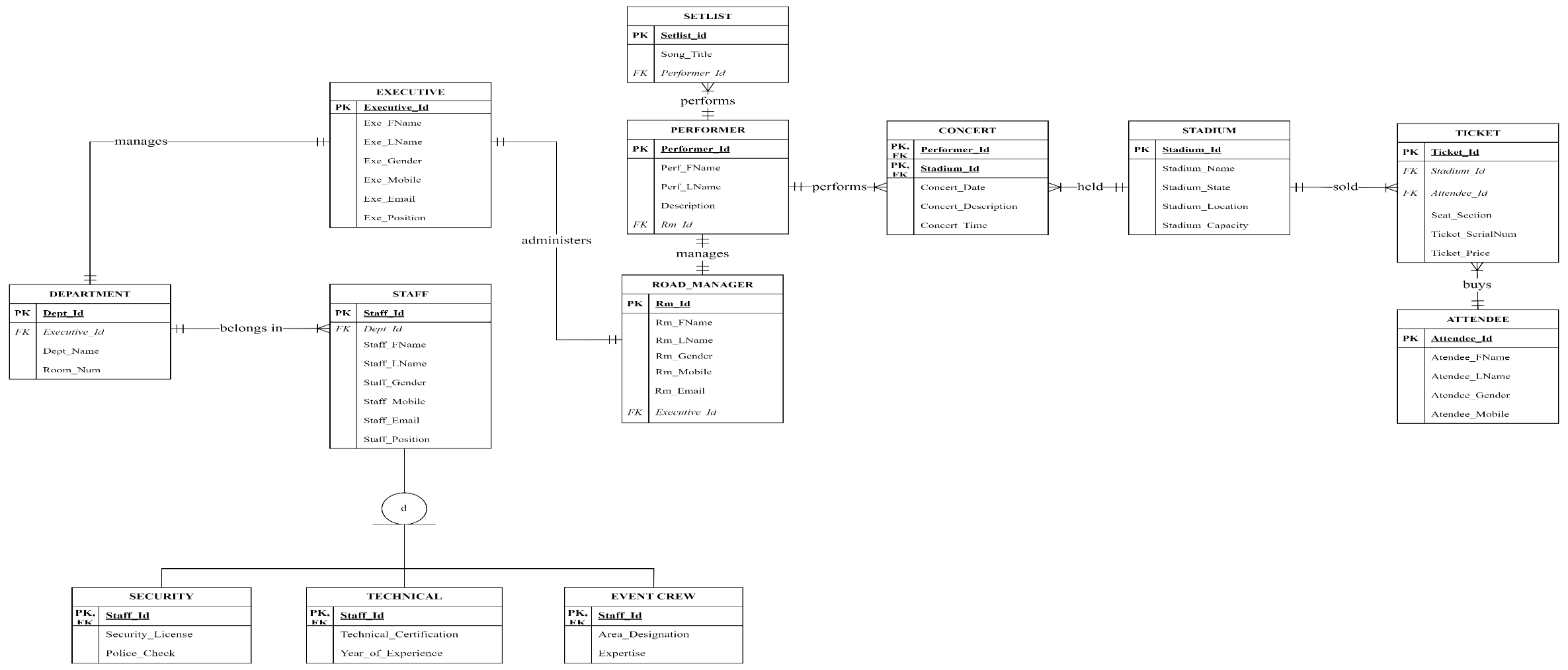


Figure ‑ BLM EERD

Figure 7-1 BLM EERD is disjoint because it shows three separate tables with its primary key (PK). A disjoint inheritance means a subclass cannot belong to more than one superclass. In this case, the tables "Security", "Technical", and "Event Crew" are all disjoint subclasses of the "Staff" superclass. A staff member can only be one type of security, technical, or event crew member.

The diagram is also partial because the three depicted subclasses do not cover all "Staff" class instances. There might be other types of staff not represented, like medical or other members, that helped to make the event a success, leaving the possibility of future subclasses to capture those categories.

# CREATION OF TABLES AND DATA POPULATION

## TABLE CREATION

* 1. Executive Table Creation

**CREATE TABLE EXECUTIVE** (

Executive\_Id VARCHAR(50) NOT NULL PRIMARY KEY ,

Exe\_FName CHAR(50) NOT NULL,

Exe\_LName CHAR(50) NOT NULL,

Exe\_Gender CHAR(1) NOT NULL,

Exe\_Mobile VARCHAR(50) NOT NULL,

Exe\_Email VARCHAR(50) NOT NULL,

Exe\_Position CHAR(50) NOT NULL

);

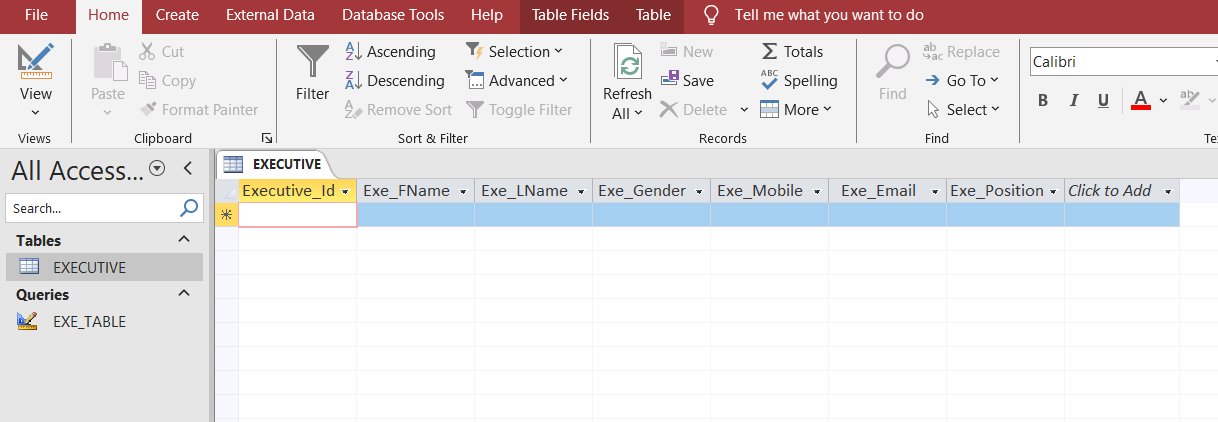
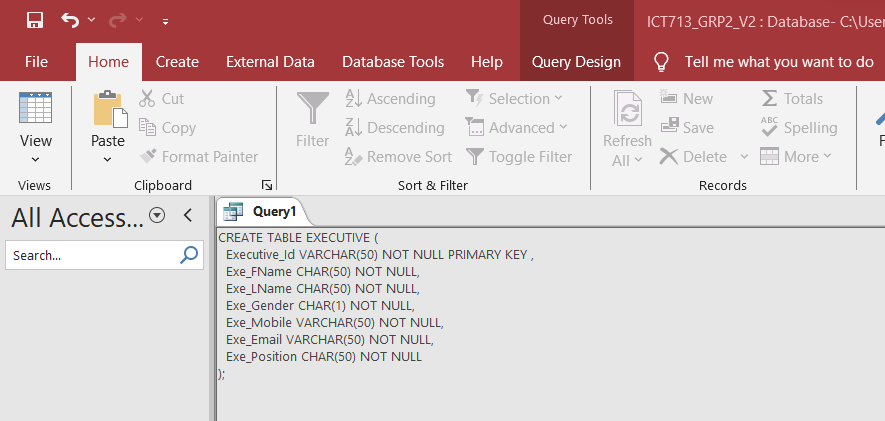


Figure ‑ Executive Table

* 1. Department Table Creation

**CREATE TABLE DEPARTMENT** (

Dept\_Id VARCHAR(50) NOT NULL PRIMARY KEY,

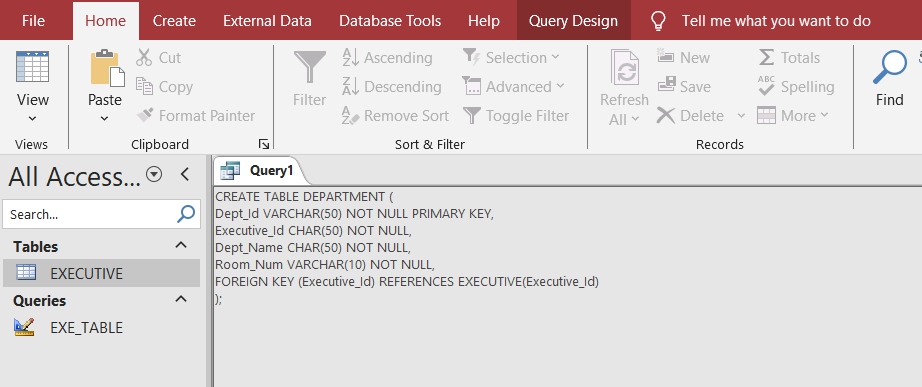
Executive\_Id CHAR(50) NOT NULL,

Dept\_Name CHAR(50) NOT NULL,

Room\_Num VARCHAR(10) NOT NULL,

FOREIGN KEY (Executive\_Id) REFERENCES EXECUTIVE(Executive\_Id)

);

**

**

Figure ‑ Department Table

* 1. Staff Table Creation

CREATE TABLE STAFF (

Staff\_Id varchar(50) NOT NULL PRIMARY KEY,

Dept\_Id char(50) NOT NULL,

Staff\_Fname char(50) NOT NULL,

Staff\_Lname char(50) NOT NULL,

Staff\_Gender char(50) NOT NULL,

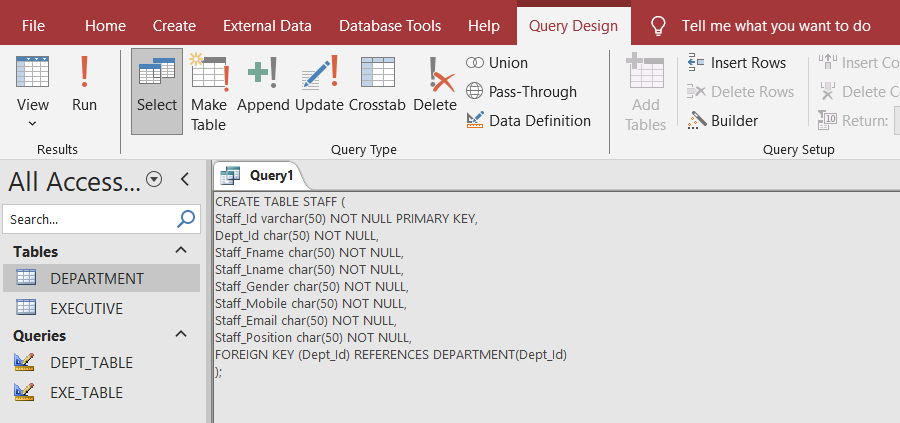
Staff\_Mobile char(50) NOT NULL,

Staff\_Email char(50) NOT NULL,

Staff\_Position char(50) NOT NULL,

FOREIGN KEY (Dept\_Id) REFERENCES DEPARTMENT(Dept\_Id)

);

****

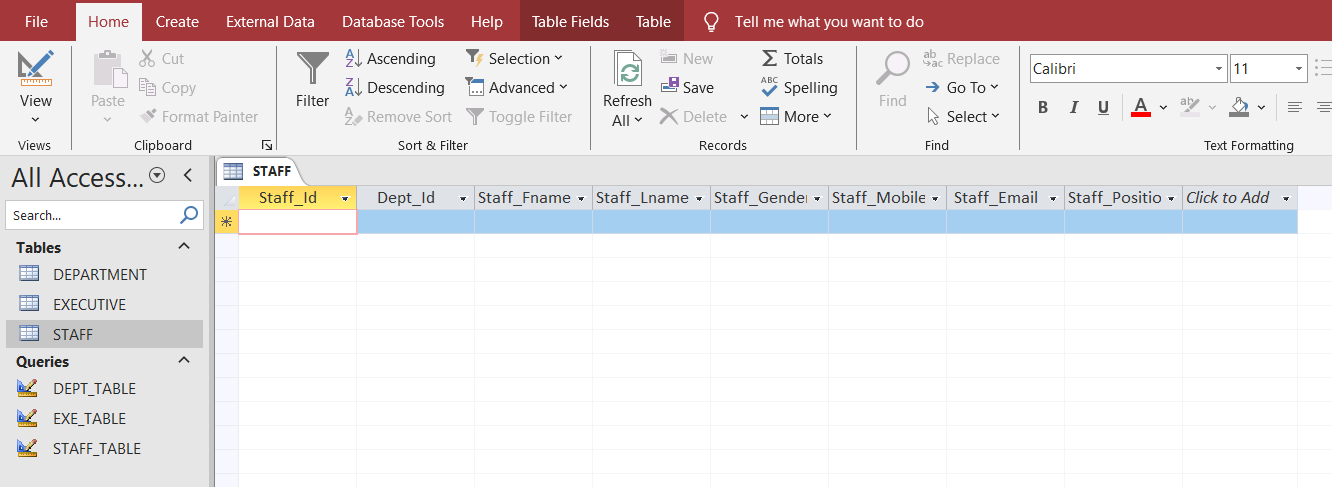
****

Figure ‑ Staff Table

* 1. Road Manager Table Creation

CREATE TABLE ROAD\_MANAGER (

Rm\_Id varchar(50) NOT NULL PRIMARY KEY,

Rm\_FName char(50) NOT NULL,

Rm\_LName char(50) NOT NULL,

Rm\_Gender char(1) NOT NULL,

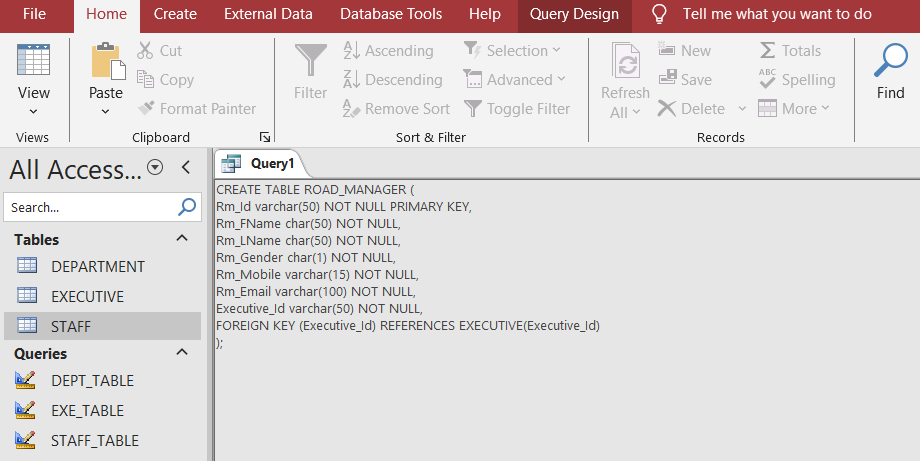
Rm\_Mobile varchar(15) NOT NULL,

Rm\_Email varchar(100) NOT NULL,

Executive\_Id varchar(50) NOT NULL,

FOREIGN KEY (Executive\_Id) REFERENCES EXECUTIVE(Executive\_Id)

);

****

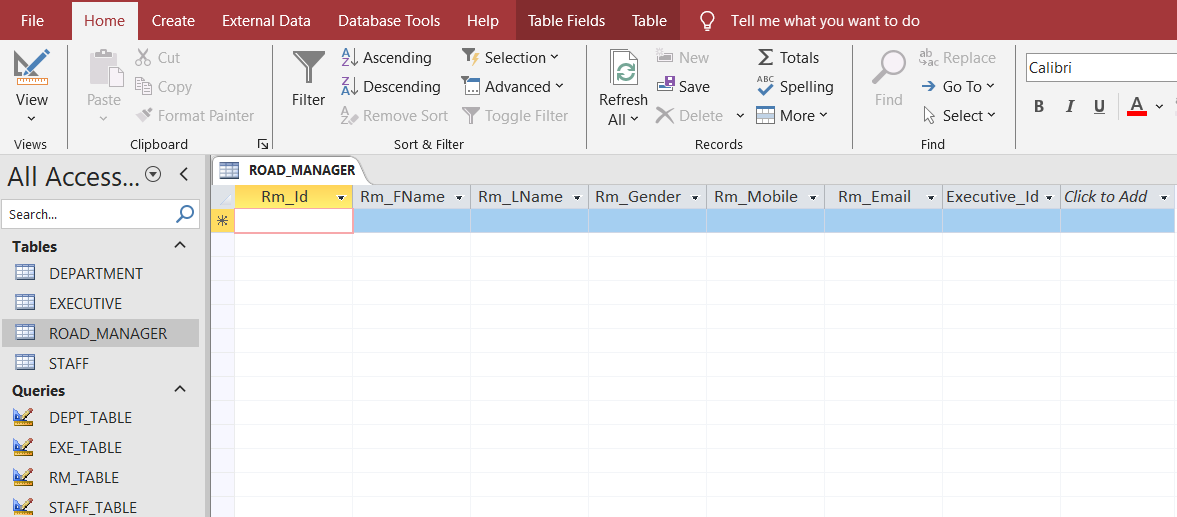
****

Figure ‑ Road Manager Table

* 1. Performer Table Creation

CREATE TABLE PERFORMER (

Performer\_Id varchar(50) NOT NULL PRIMARY KEY,

Perf\_FName char(50) NOT NULL,

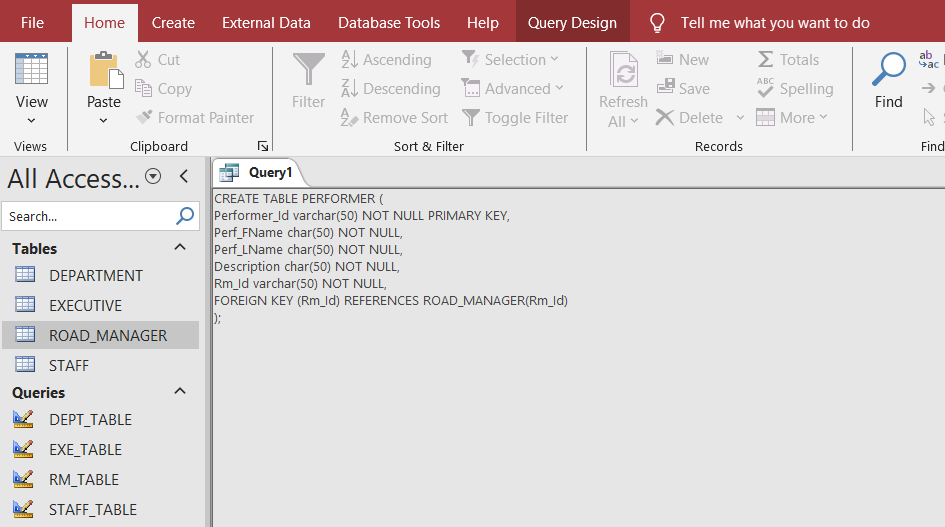
Perf\_LName char(50) NOT NULL,

Description char(50) NOT NULL,

Rm\_Id varchar(50) NOT NULL,

FOREIGN KEY (Rm\_Id) REFERENCES ROAD\_MANAGER(Rm\_Id)

);

****

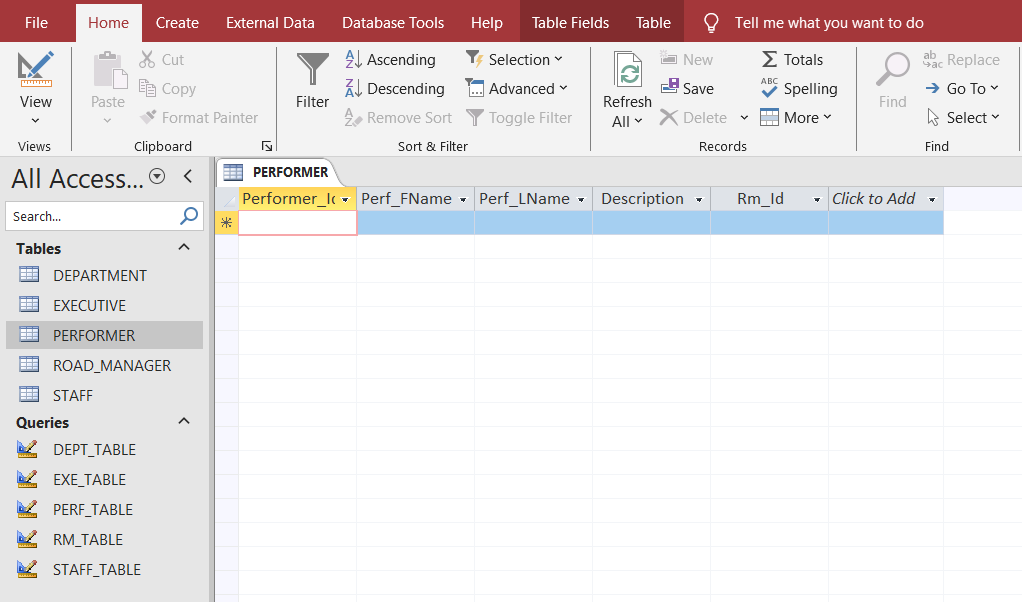
****

Figure ‑ Performer Table

* 1. Setlist Table Creation

CREATE TABLE SETLIST (

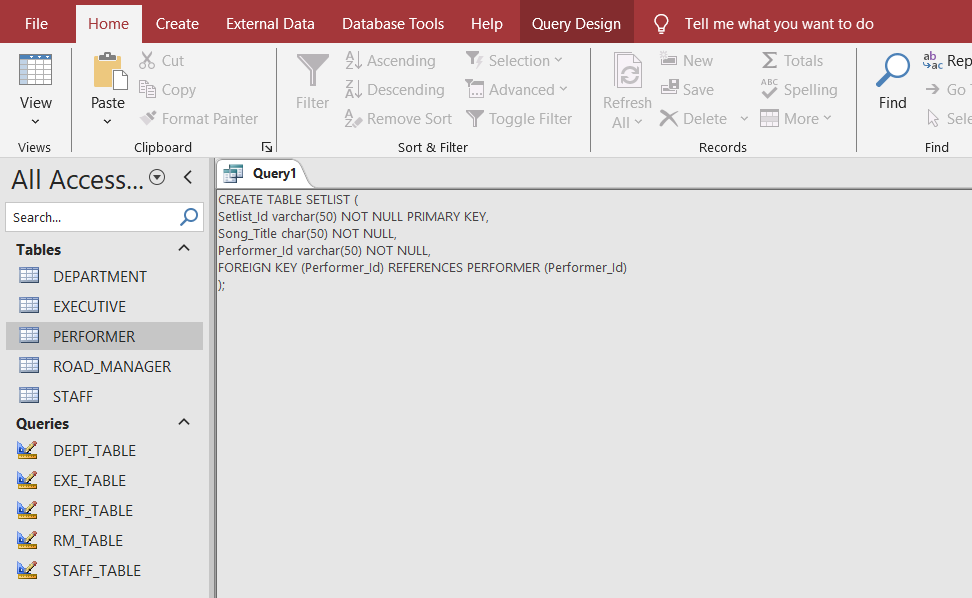
Setlist\_Id varchar(50) NOT NULL PRIMARY KEY,

Song\_Title char(50) NOT NULL,

Performer\_Id varchar(50) NOT NULL,

FOREIGN KEY (Performer\_Id) REFERENCES PERFORMER (Performer\_Id)

);

****

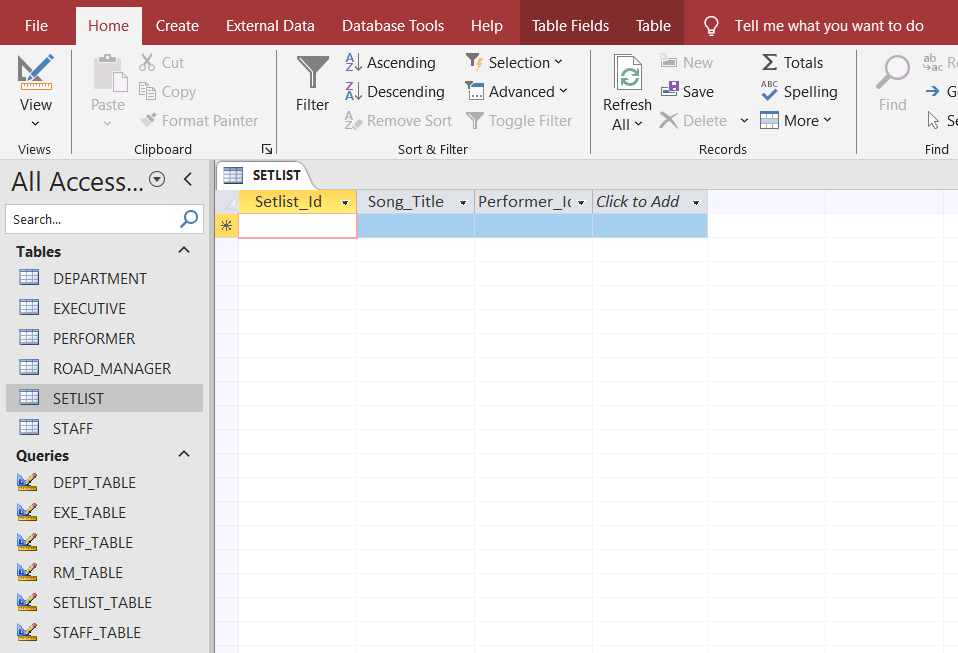
****

Figure ‑ Setlist Table

* 1. Stadium Table Creation

CREATE TABLE STADIUM (

Stadium\_Id varchar(50) NOT NULL PRIMARY KEY,

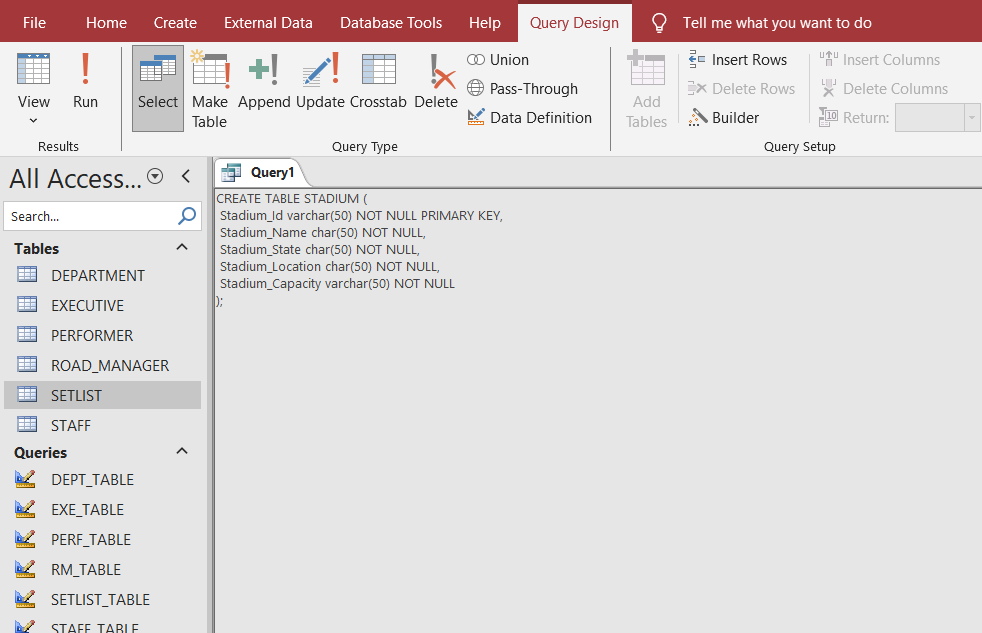
Stadium\_Name char(50) NOT NULL,

Stadium\_State char(50) NOT NULL,

Stadium\_Location char(50) NOT NULL,

Stadium\_Capacity varchar(50) NOT NULL

);



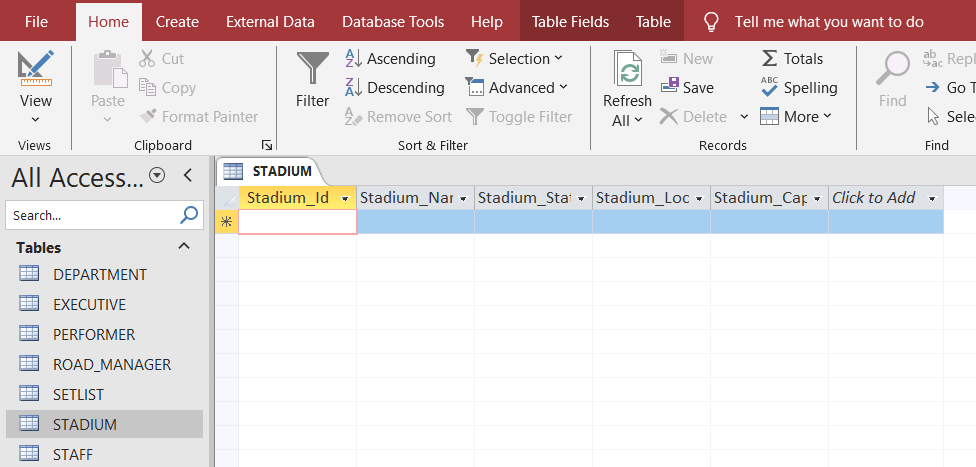


Figure ‑ Stadium Table

* 1. Concert Table Creation

CREATE TABLE CONCERT (

Performer\_Id varchar(50) NOT NULL,

Stadium\_Id varchar(50) NOT NULL,

Concert\_Date DATE NOT NULL,

Concert\_Description char(50) NOT NULL,

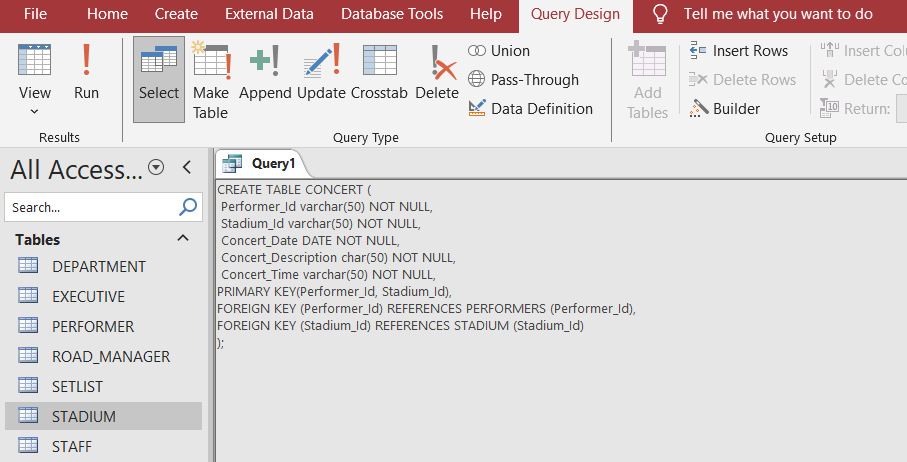
Concert\_Time varchar(50) NOT NULL,

PRIMARY KEY(Performer\_Id, Stadium\_Id),

FOREIGN KEY (Performer\_Id) REFERENCES PERFORMER (Performer\_Id),

FOREIGN KEY (Stadium\_Id) REFERENCES STADIUM (Stadium\_Id)

);

****

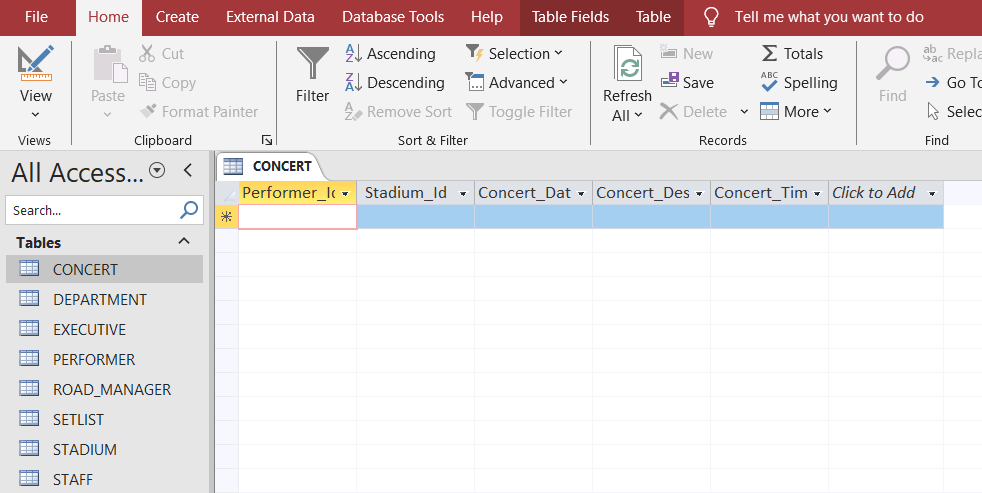
****

Figure ‑ Concert Table

* 1. Attendee Table Creation

CREATE TABLE ATTENDEE (

Attendee\_Id varchar(50) NOT NULL PRIMARY KEY,

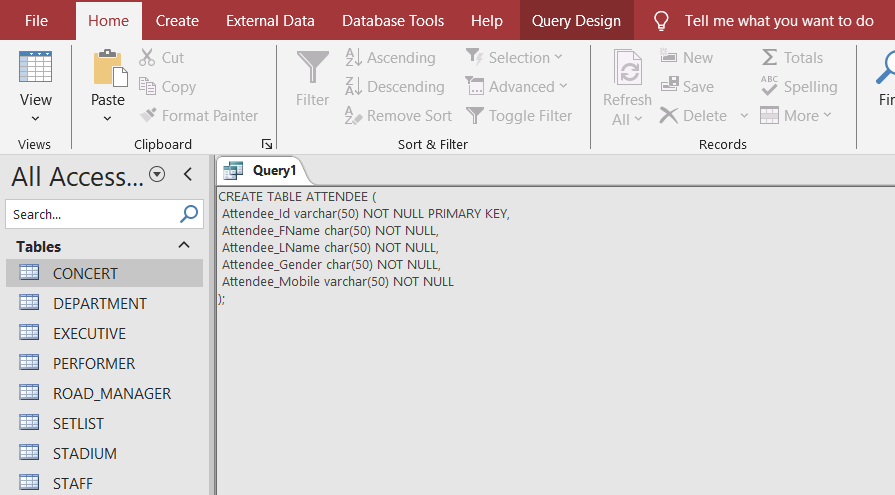
Attendee\_FName char(50) NOT NULL,

Attendee\_LName char(50) NOT NULL,

Attendee\_Gender char(50) NOT NULL,

Attendee\_Mobile varchar(50) NOT NULL

);

****

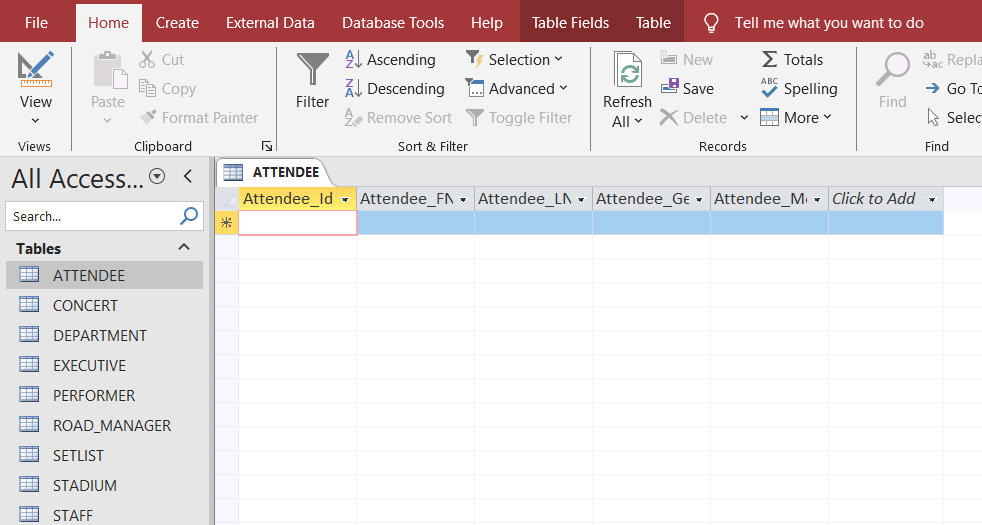
****

Figure ‑ Attendee Table

* 1. Ticket Table Creation

CREATE TABLE TICKET (

Ticket\_Id varchar(50) NOT NULL PRIMARY KEY,

Stadium\_Id varchar(50) NOT NULL,

Attendee\_Id varchar(50) NOT NULL,

Seat\_Section varchar(50) NOT NULL,

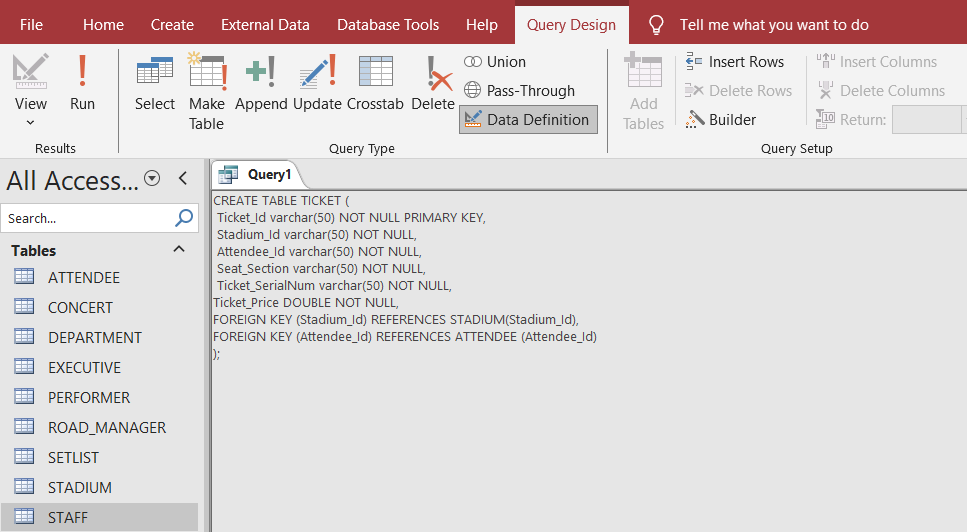
Ticket\_SerialNum varchar(50) NOT NULL,

Ticket\_Price DOUBLE NOT NULL,

FOREIGN KEY (Stadium\_Id) REFERENCES STADIUM(Stadium\_Id),

FOREIGN KEY (Attendee\_Id) REFERENCES ATTENDEE (Attendee\_Id)

);



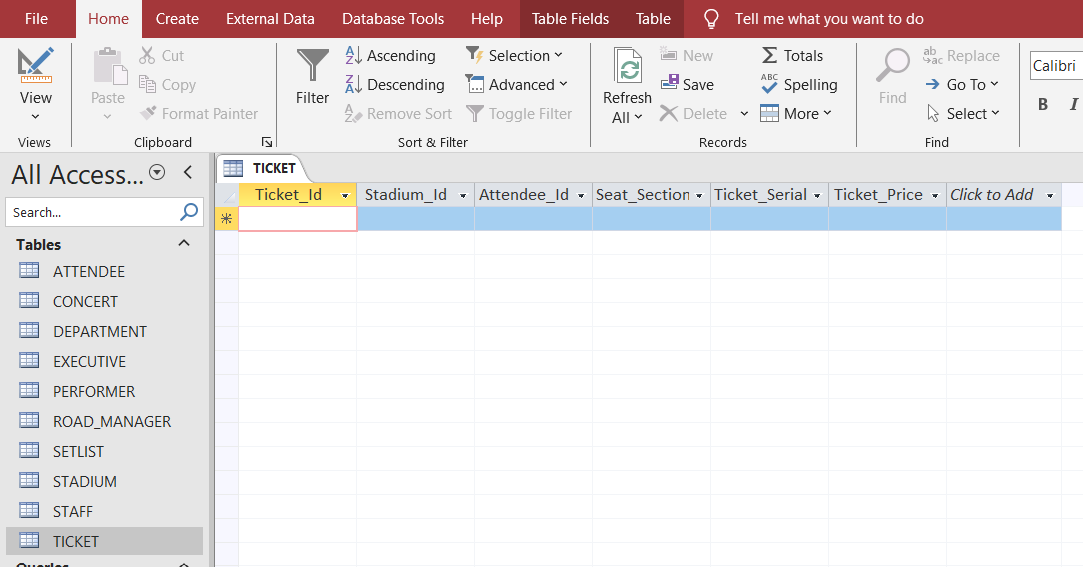


Figure ‑ Ticket Table

* 1. Security Table Creation

CREATE TABLE SECURITY (

Staff\_Id varchar(50) NOT NULL PRIMARY KEY,

Security\_License varchar(50) NOT NULL,

Police\_Check varchar(50) NOT NULL,

FOREIGN KEY (Staff\_Id) REFERENCES STAFF(Staff\_Id)

);

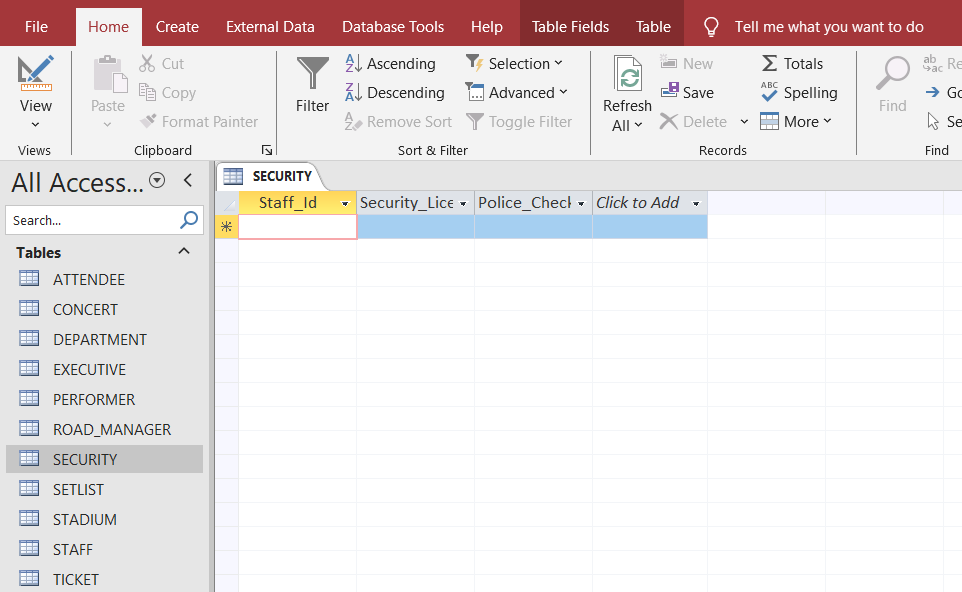
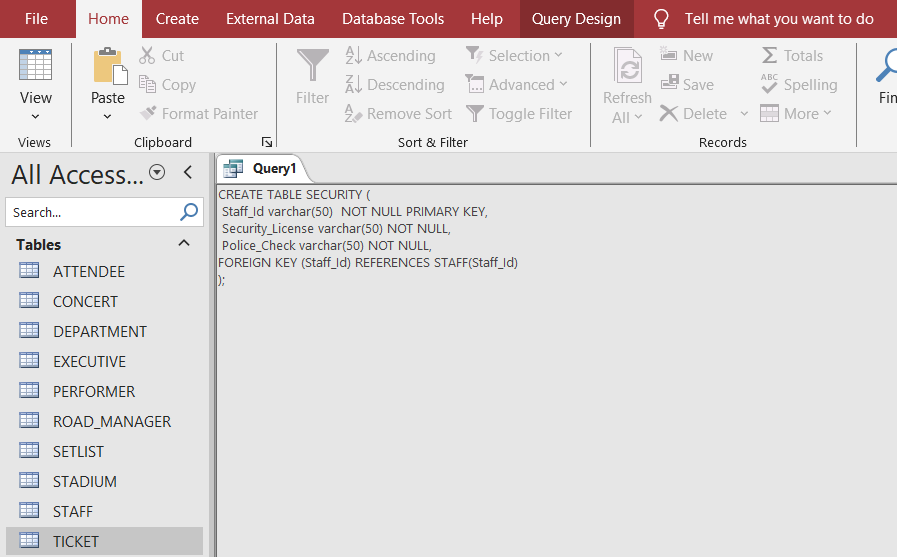


Figure ‑ Security Table

* 1. Technical Table Creation

CREATE TABLE TECHNICAL (

Staff\_Id varchar(50) NOT NULL PRIMARY KEY,

Technical\_Certification varchar(50) NOT NULL,

Years\_of\_Experience INT NOT NULL,

FOREIGN KEY (Staff\_Id) REFERENCES STAFF(Staff\_Id)

);

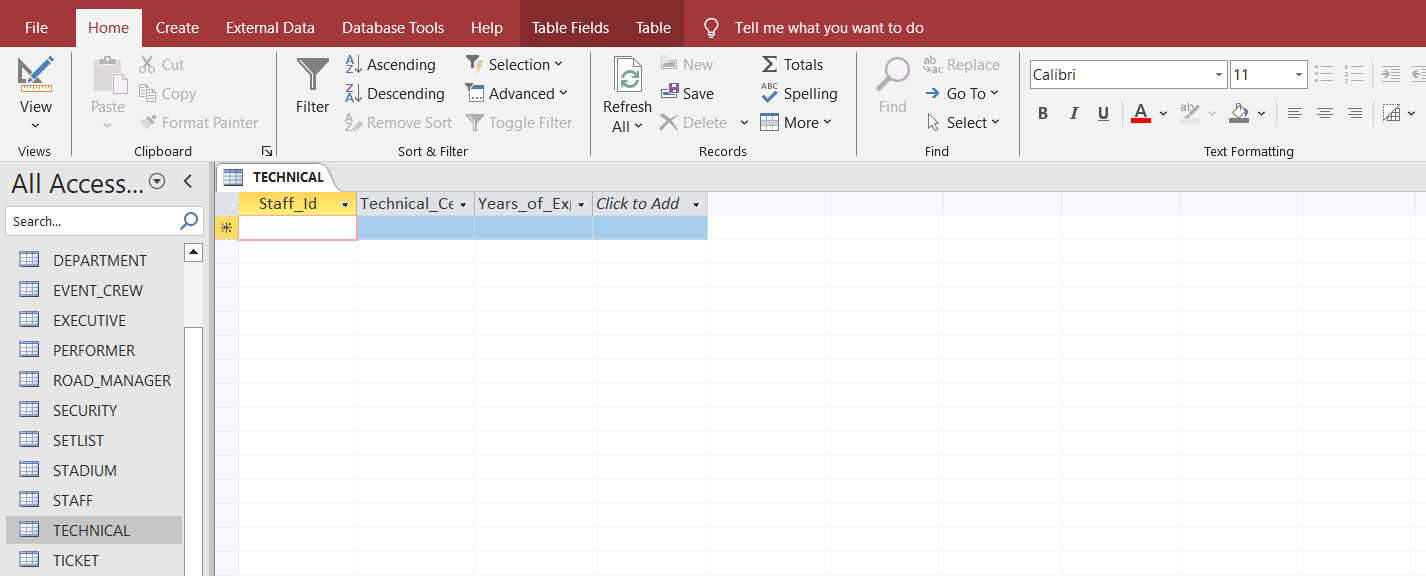
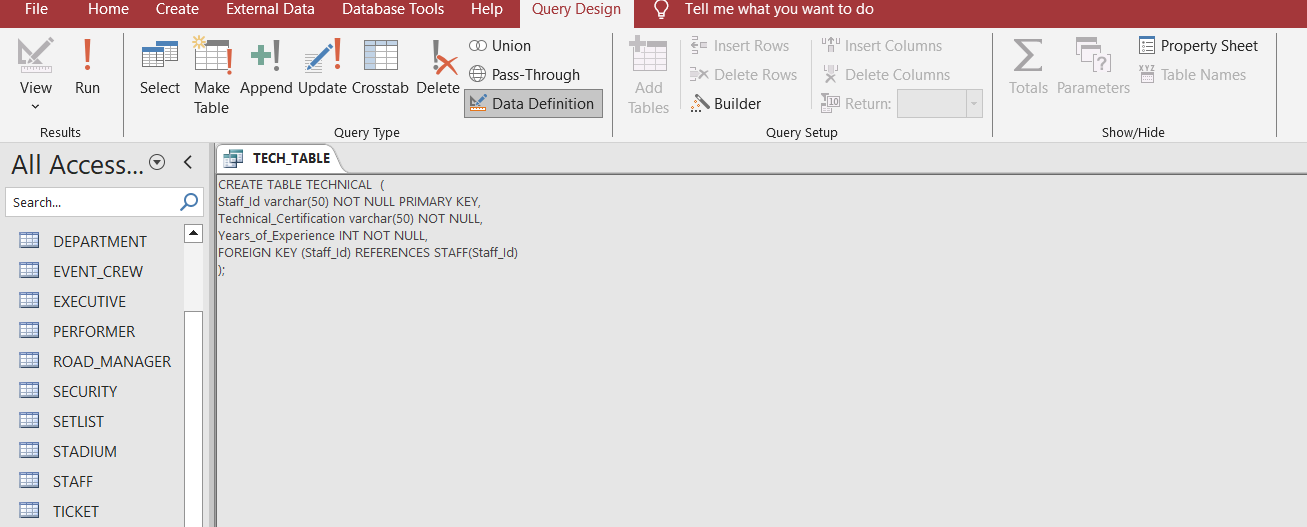


Figure ‑ Technical Table

* 1. Event\_Crew Table Creation

CREATE TABLE EVENT\_CREW (

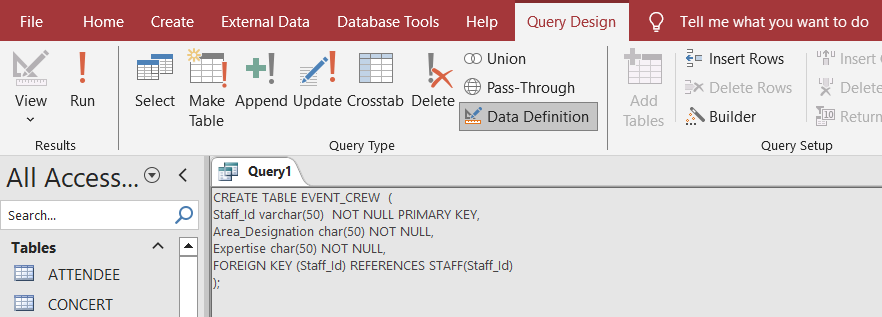
Staff\_Id varchar(50) NOT NULL PRIMARY KEY,

Area\_Designation char(50) NOT NULL,

Expertise char(50) NOT NULL,

FOREIGN KEY (Staff\_Id) REFERENCES STAFF(Staff\_Id)

);



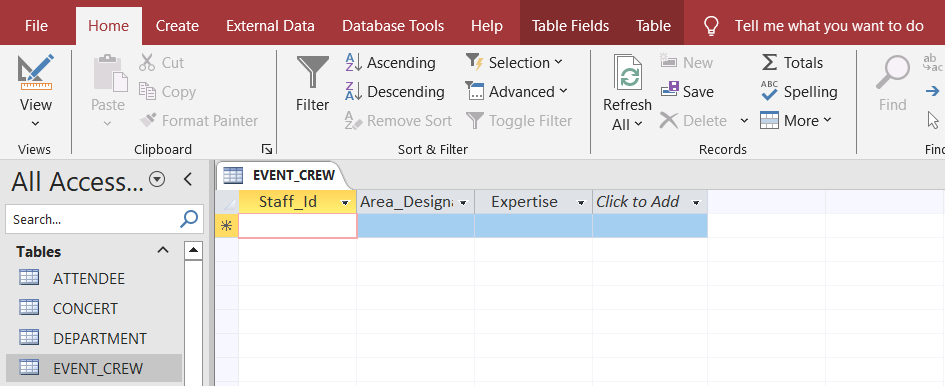


Figure ‑ Event\_Crew Table

## DATA POPULATION

* + - * 1. Insert Executive Records

INSERT INTO EXECUTIVE(Executive\_Id, Exe\_FName, Exe\_LName, Exe\_Gender, Exe\_Mobile, Exe\_Email, Exe\_Position)

VALUES

("BMLEX101","ANDREA","MILLER","F","425465485","user1@gmail.com","PAX Manager"),

("BMLEX102","BRYAN","COOPER","M","412658794","user2@gmail.com","TECT Manager"),

("BMLEX103","STEPHANIE","JONES","F","415369874","user3@gmail.com","ADS Manager"),

("BMLEX104","ANDREW","MARTINES","M","402156486","user4@gmail.com","TIC Manager"),

("BMLEX105","BARONE","LEE","M","420156975","user5@gmail.com","SAS Manager");

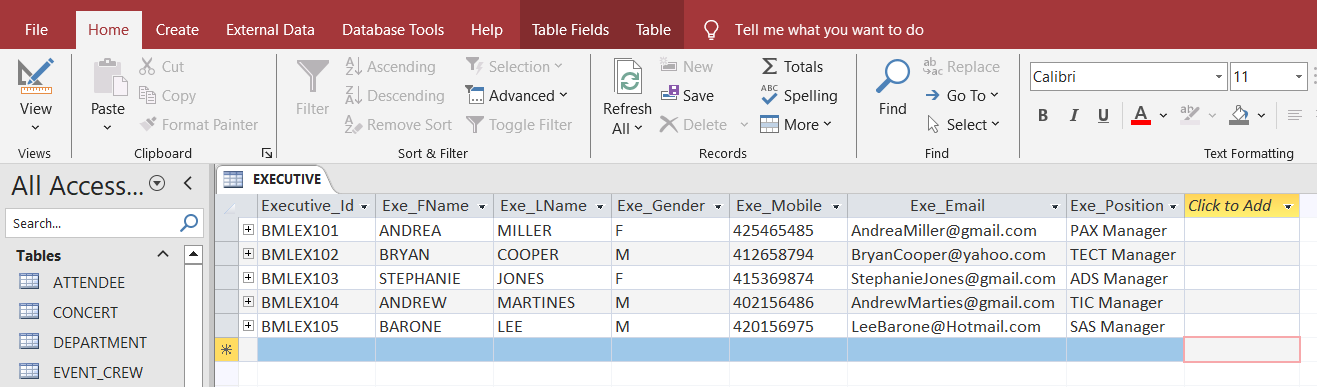
****

Figure ‑ Executive Records

* + - * 1. Insert Department Records

INSERT INTO DEPARTMENT (Dept\_Id, Executive\_Id, Dept\_Name, Room\_Num)

VALUES

("BMLPAX001","BMLEX101","Planning and Execution","101"),

("BMLTECT001","BMLEX102","Technical Team","102"),

("BMLADS001","BMLEX103","Advertisment and Promotion","103"),

("BMLTIC001","BMLEX104","Ticketing Department","104"),

("BMLSAS001","BMLEX105","Security and Safety","105");

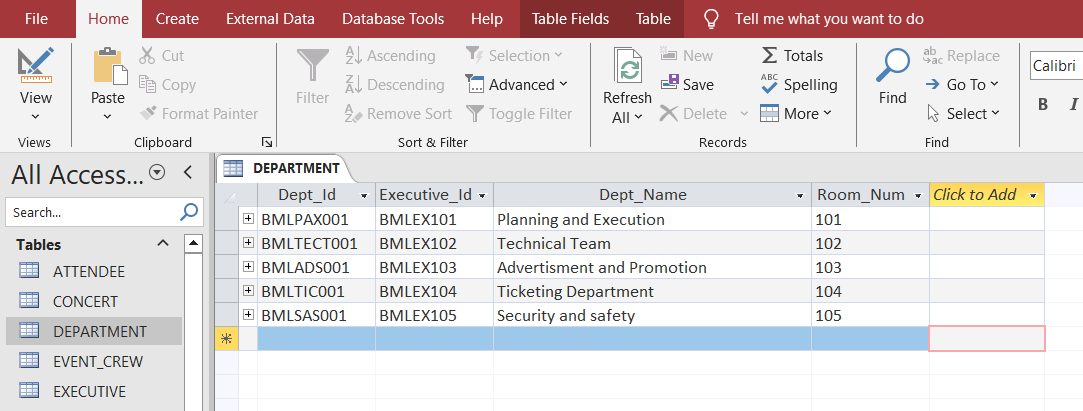
****

Figure ‑ Department Records

* + - * 1. Insert Road Manager Records

INSERT INTO ROAD\_MANAGER (Rm\_Id, Rm\_FName, Rm\_LName, Rm\_Gender, Rm\_Mobile, Rm\_Email, Executive\_Id )

VALUES

("BMLRM411","Miranda","Sprite","F",”415486111”,”rmuser1@gmail.com”,”BMLEX101”),

("BMLRM412","Jericho","Rosales","M",”412587463”,”rmuser2@gmail.com”,”BMLEX102”),

("BMLRM413","Daniel","Padilla","F",”425795843”,”rmuser3@gmail.com”,”BMLEX103”),

("BMLRM414","Kathryn,"Bernardo,"F",”425169873”,”rmuser4@gmail.com”,”BMLEX104”),

("BMLRM415","Sandara","Park","F",”425486548”,”[rmuser5@gmail.com](mailto:rmuser5@gmail.com)”,”BMLEX105”);

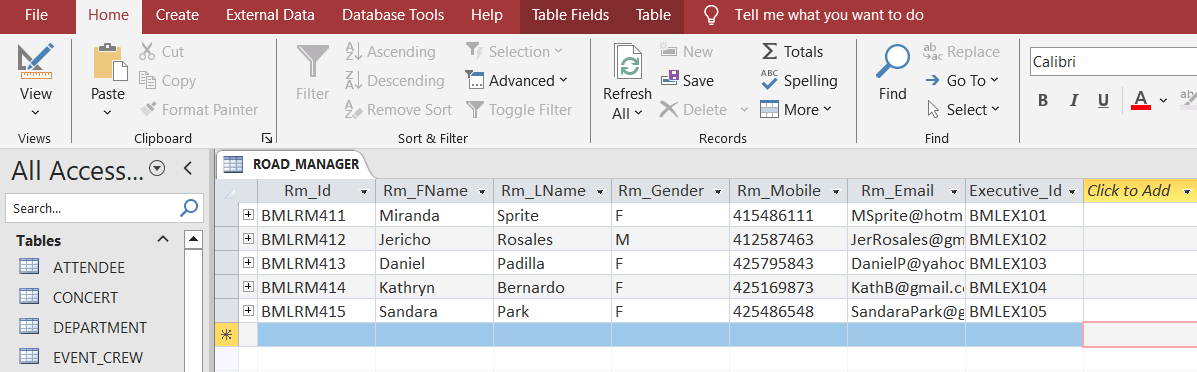
****

Figure ‑ Road Manager Records

* + - * 1. Insert Performer Records

INSERT INTO PERFORMERS (Performer\_id, Perf\_FName, Perf\_LName, Description, Rm\_Id )

VALUES

("BMLPERF511","Bruno","Mars","Main Act",”BMLRM411”),

("BMLPERF512","Spinall","Nigeria","Guest Performer",”BMLRM412”),

("BMLPERF513","Lewis","Capaldi","Special Guest",”BMLRM413”),

("BMLPERF514","Paul","Klein","Band Member",”BMLRM414”),

("BMLPERF515","Kris","Roberts","Band Member",”BMLRM415”);

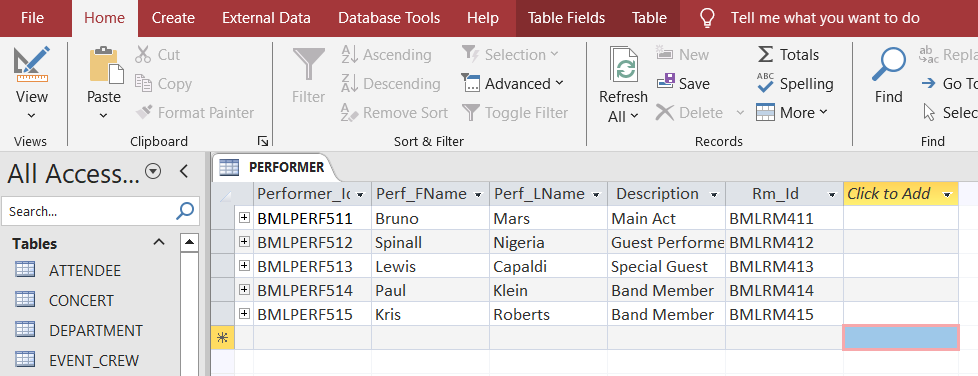
****

Figure ‑ Performer Records

* + - * 1. Insert Setlist Records

INSERT INTO SETLIST (Setlist\_Id, Song\_Title, Performer\_Id) VALUES

("BMLSET811",”Talking to the moon","BMLPERF511"),

("BMLSET812",”Versace on the floor","BMLPERF512"),

("BMLSET813",”Leave the door open","BMLPERF513"),

("BMLSET814",”Sere","BMLPERF514"),

("BMLSET815",”Before you go","BMLPERF515");

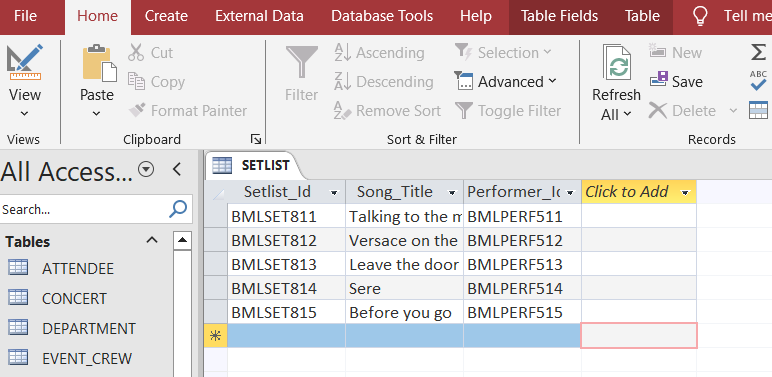
****

Figure ‑ Setlist Records

* + - * 1. Insert Stadium Records

INSERT INTO STADIUM (Stadium\_Id, Stadium\_Name, Stadium\_State, Stadium\_Location, Stadium\_Capacity)

VALUES

("BMLSYD811",”OPERA HOUSE CONCERT HALL","SYDNEY”, “Bennelong Point, Sydney NSW 200”, “5000”),

("BMLMEL812",”ROD LAVER ARENA","MELBOURNE”, “Olympic Blvd, Melbourne VIC 3001”, “6000”),

("BMLHBF813",”HBF STADIUM","PERTH”, “100 Stephenson Ave, Mount Claremont WA 6010”, “4000”),

("BMLADL814",”ADELAIDE OVAL","ADELAIDE”, “War Memorial Dr, North Adelaide SA 5006”, “4500”),

("BMLBRIS15",”BRISBANE ENTERTAINMENT CENTRE","BRISBANE”, “1 Melaleuca Dr, Boondall QLD 4034”, “4000”);

****

Figure ‑ Stadium Records

* + - * 1. Insert Attendee Records

INSERT INTO ATTENDEE (**Attendee\_Id**, Attendee\_FName, Attendee\_LName, Attendee\_Gender, Attendee\_Mobile)

VALUES

("BMLSYD1011",”PAULO","ASTRA”, “M”, “415469875”),

("BMLSYD1012",”VIC","POTTER”, “M”, “421251212”),

("BMLSYD1013",”JURYCAH","SANTOS”, “F”, “456215478”),

("BMLSYD1014",”EDSAR","GAMBOA”, “M”, “400001125”),

("BMLSYD1015",”JASPER","DON”, “M”, “402132658”),

("BMLMEL1011",”GARY",PENA”, “M”, “478958685”),

("BMLMEL1012",”PETER","DAYRIT”, “M”, “412566565”),

("BMLMEL1013",”ANALYN","COPPER”, “F”, “422222565”),

("BMLMEL1014",”PILAR","BLACK”, “F”, “411565489”),

("BMLMEL1015",”ROBERT","LORD”, “M”, “421353655”),

("BMLHBF1011",”MARK","CUDAL”, “M”, “403256556”),

("BMLHBF1012",”BRYAN","RAMIREZ”, “M”, “401215478”),

("BMLHBF1013",”SHAI","ESCOTO”, “F”, “412365849”),

("BMLHBF1014",”DENVER","GARCIA”, “M”, “402151214”),

("BMLHBF1015",”TRIXIA","BATUL”, “F”, “401112212”),

("BMLADL1011",”LANCE","LORD”, “M”, “459864789”),

("BMLADL1012",”JAZZ","CARVEL”, “M”, “455566688”),

("BMLADL1013",”JAMES","PEDRINA”, “M”, “451214566”),

("BMLADL1014",”ROSS","LAXA”, “M”, “403256545”),

("BMLADL1015",”JOMAR","MANDAP”, “M”, “462153548”),

("BMLBRIS1011",”JURYCAH","BENNY”, “F”, “465848754”),

("BMLBRIS1012",”EDSAR","CORPUS”, “M”, “451324111”),

("BMLBRIS1013",”JASPER","DIAS”, “M”, “421548845”),

("BMLBRIS1014",”RENEL","CRUZ”, “M”, “421454114”),

("BMLBRIS1015",”CHRISTINE","PEREZ”, “M”, “412154444”);

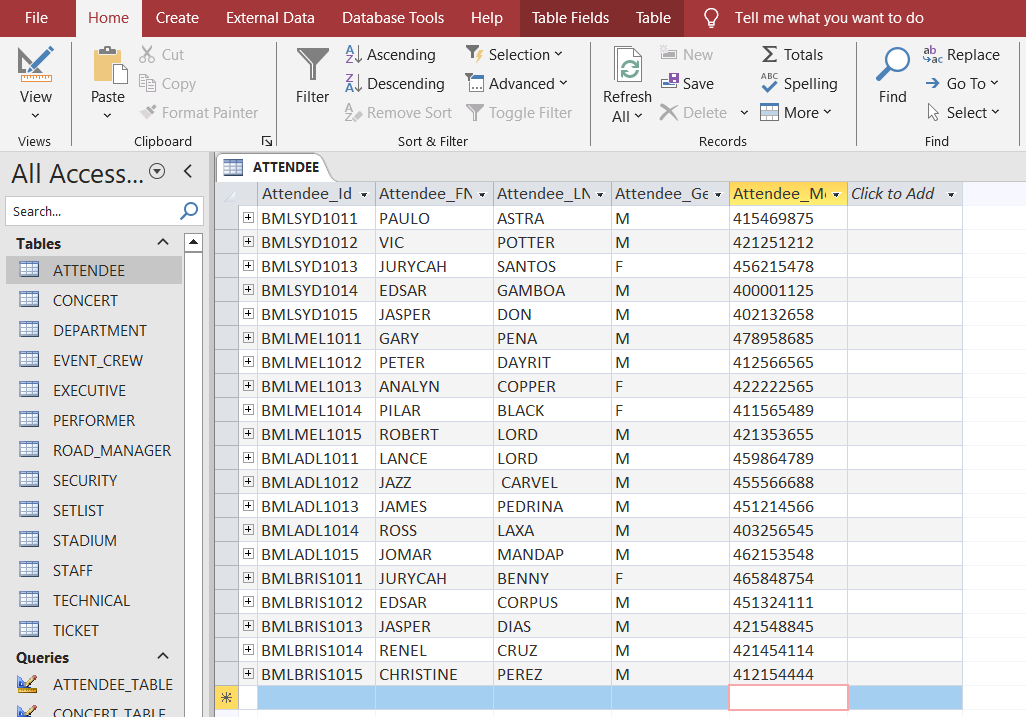
****

Figure ‑ Attendee Records

* + - * 1. Insert Ticket Records

INSERT INTO TICKET (Ticket\_Id, Stadium\_Id, Attendee\_Id, Seat\_Section, Ticket\_SerialNum, Ticket\_Price)

VALUES

("BMLSYD711",”BMLSYD811","BMLSYD1011”, “VIP FRONT”, “BSOS01”,”280”),

("BMLSYD712",”BMLSYD811","BMLSYD1012”, “VIP FRONT”, “BSOS02”,”280”),

("BMLSYD713",”BMLSYD811","BMLSYD1013”, “MIDDLE”, “BSOS03”,”250”),

("BMLSYD714",”BMLSYD811","BMLSYD1014”, “MIDDLE”, “BSOS04”,”250”),

("BMLSYD715",”BMLSYD811","BMLSYD1015”, “BACK”, “BSOS05”,”200”),

("BMLMEL711",”BMLMEL812","BMLMEL1011”, “VIP FRONT”, “BMOS01”,”280”),

("BMLMEL712",”BMLMEL812","BMLMEL1012”, “VIP FRONT”, “BMOS02”,”280”),

("BMLMEL713",”BMLMEL812","BMLMEL1013”, “MIDDLE”, “BMOS03”,”250”),

("BMLMEL714",”BMLMEL812","BMLMEL1014”, “MIDDLE”, “BMOS04”,”250”),

("BMLMEL715",”BMLMEL812","BMLMEL1015”, “BACK”, “BMOS05”,”200”),

("BMLPRH711",”BMLHBF813","BMLHBF1011”, “VIP FRONT”, “BPOS01”,”280”),

("BMLPRH712",”BMLHBF813","BMLHBF1012”, “VIP FRONT”, “BPOS02”,”280”),

("BMLPRH713",”BMLHBF813","BMLHBF1013”, “MIDDLE”, “BPOS03”,”250”),

("BMLPRH714",”BMLHBF813","BMLHBF1014”, “MIDDLE”, “BPOS04”,”250”),

("BMLPRH715",”BMLHBF813","BMLHBF1015”, “BACK”, “BPOS05”,”200”),

("BMLADL11",”BMLADL814","BMLADL1011”, “VIP FRONT”, “BAOS01”,”280”),

("BMLADL12",”BMLADL814","BMLADL1012”, “VIP FRONT”, “BAOS02”,”280”),

("BMLADL13",”BMLADL814","BMLADL1013”, “MIDDLE”, “BAOS03”,”250”),

("BMLADL14",”BMLADL814","BMLADL1014”, “MIDDLE”, “BAOS04”,”250”),

("BMLADL15",”BMLADL814","BMLADL1015”, “BACK”, “BAOS05”,”200”),

("BMLBRIS11",”BMLBRIS15","BMLBRIS1011”, “VIP FRONT”, “BBOS01”,”280”),

("BMLBRIS12",”BMLBRIS15","BMLBRIS1012”, “VIP FRONT”, “BBOS02”,”280”),

("BMLBRIS13",”BMLBRIS15","BMLBRIS1013”, “MIDDLE”, “BBOS03”,”250”),

("BMLBRIS14",”BMLBRIS15","BMLBRIS1014”, “MIDDLE”, “BBOS04”,”250”),

("BMLBRIS15",”BMLBRIS15","BMLBRIS1015”, “BACK”, “BBOS05”,”200”);

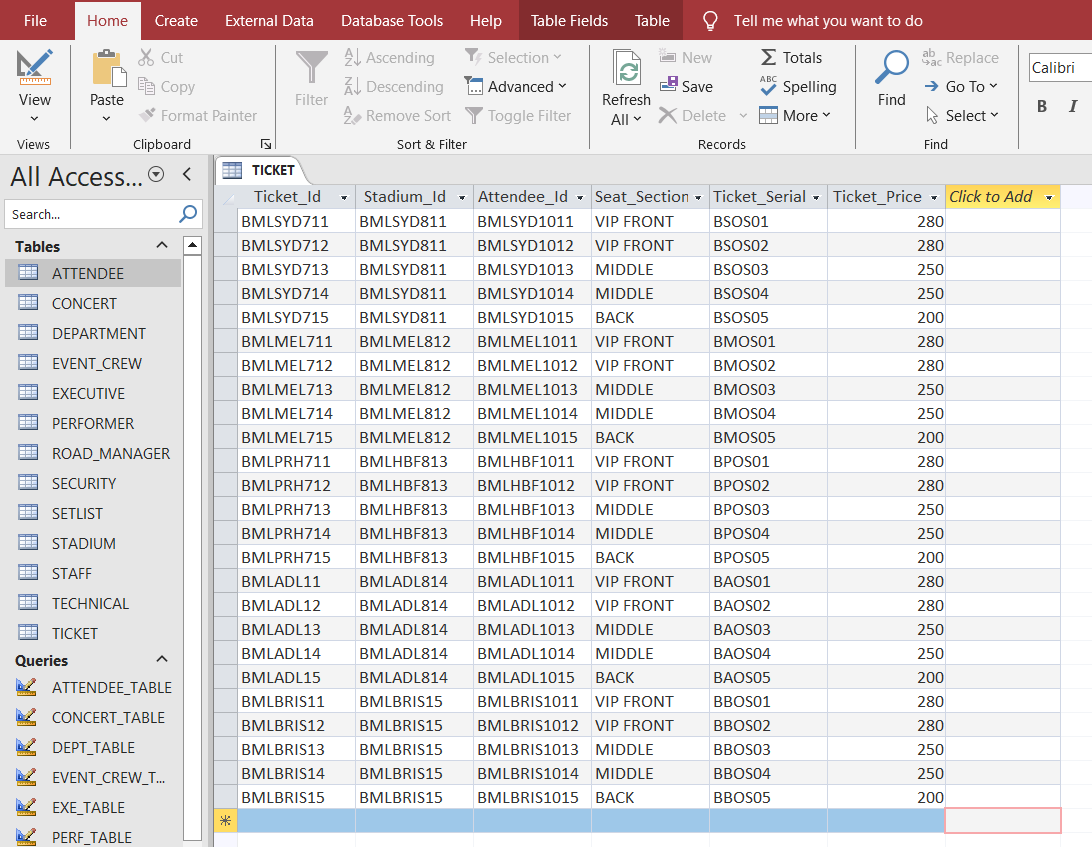
****

Figure ‑ Ticket Records

* + - * 1. Insert Concert Records

INSERT INTO CONCERT(Performer\_Id, Stadium\_Id, Concert\_Date, Concert\_Description, Concert\_Time)

VALUES

("BMLPERF511","BMLSYD811","14/10/2023","Day 1",”1700”),

("BMLPERF512","BMLMEL812","15/10/2023","Day 2",”1800”),

("BMLPERF513","BMLHBF813","21/10/2023","Day 3",”1700”),

("BMLPERF514","BMLADL814","22/10/2023","Day 4",”1700”),

("BMLPERF515","BMLBRIS15","28/10/2023","Day 5",”1800”);

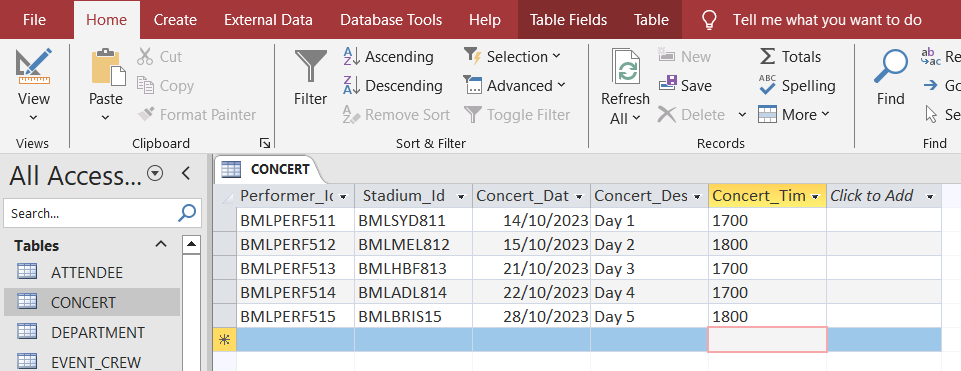
****

Figure ‑ Concert Records

* + - * 1. Insert Staff Records

INSERT INTO STAFF (Staff\_Id, Dept\_Id, Staff\_FName, Staff\_LName, Staff\_Gender, Staff\_Mobile, Staff\_Email, Staff\_Position)

VALUES

("BMLEMP01",,"BMLPAX001","JERWIN",”MANDAP”,”M ”,”423657841”,”JerwinMAndap@gmail.com”,”PAX Staff”),

("BMLEMP02",,"BMLPAX001","KEN",”LEE”,”M”,”458368942”,”KN@gmail.com”,”PAX Staff”),

("BMLEMP03",,"BMLPAX001","CARL,”MAROLSE”,”F”,”410236589”,”CarlMar@gmail.com”,”PAX Staff”),

("BMLEMP04",,"BMLPAX001","BON",”MAKER”,”F”,”412587963”,”BonMaker@gmail.com”,”PAX Staff”),

("BMLEMP05",,"BMLPAX001","JAZZ,”MILLER”,”M”,”402578946”,”Jazz@gmail.com”,”PAX Staff”),

("BMLEMP06",,"BMLTECT001","CATHERINE",”LAXA ”,”F”,”475894215”,”Catherine@gmail.com ”,”TECHNICAL”),

("BMLEMP07",,"BMLTECT001","ANOTHONY",”GOOSE”,”M”,”425798156”,”AntonGoose@gmail.com”,”TECHNICAL”),

("BMLEMP08",,"BMLTECT001","MARK",”CHUA”,”M”,”413658794”,”ChuaMark@gmail.com”,”TECHNICAL”),

("BMLEMP09",,"BMLTECT001","REY",”BUAN”,”M”,”436594824”,”ReyBuan@gmail.com”,”TECHNICAL”),

("BMLEMP10",,"BMLTECT001","ARNOLD",”WATTS”,”M”,”412035879”,”WattsArn@gmail.com”,”TECHNICAL”),

("BMLEMP11",,"BMLADS001","JOMAR",”PEDRINA ”,”M”,”416549785”,”Joms@gmail.com”,”EVENT CREW”),

("BMLEMP12",,"BMLADS001","MARTIN",”CRUZ”,”M”,”431569548”,”MartinCruz@gmail.com”,”EVENT CREW”),

("BMLEMP13",,"BMLADS001","JOHN",”WALE”,”M”,”402157594”,”JohnWale@gmail.com”,”EVENT CREW”),

("BMLEMP14",,"BMLADS001","GUARAV",”CHAT”,”M”,”401254658”,”GuaravChat@gmail.com”,”EVENT CREW”),

("BMLEMP15",,"BMLADS001","GEN",”WILL”,”F”,”412569854”,”Gw@gmail.com”,”EVENT CREW”),

("BMLEMP16",,"BMLTIC001","ROSS",”CARVEL ”,”M”,”410213265”,”RossC@gmail.com”,”TICKETING STAFF”),

("BMLEMP17",,"BMLTIC001","GREY",”BLACK”,”M”,”402154658”,”Grer123@gmail.com”,”TICKETING STAFF”),

("BMLEMP18",,"BMLTIC001","MARIE",”JACK”,”F”,”402136548”,”MarieJack02@gmail.com”,”TICKETING STAFF”),

("BMLEMP19","BMLTIC001","AMBER",”DEE”,”F”,”402318496”,”AmberDee@gmail.com”,”TICKETING STAFF”),

("BMLEMP20",,"BMLTIC001","MICHELLE",”MULAN”,”F”,”421659874”,”Mulanmich@gmail.com”,”TICKETING STAFF”),

("BMLEMP25",,"BMLSAS001","JAZZ",”BLACK ”,”M”,”458941240”,”JazzBlack@gmail.com ”,”SECURITY”),

("BMLEMP21",,"BMLSAS001","TRIXIA",”GUERRERO”,”M”,”413254942”,”TrixiaGuer@gmail.com”,”SECURITY”),

("BMLEMP22",,"BMLSAS001","LANIE",”PENA”,”M”,”402121513”,”LaniePEna@gmail.com”,”SECURITY”),

("BMLEMP23",,"BMLSAS001","LOVELY",”DAYRIT”,”M”,”401212153”,”DayritLove@gmail.com”,”SECURITY”),

("BMLEMP24",,"BMLSAS001","LANCE",”COPPER”,”M”,”415476214”,”Lance03@gmail.com”,”SECURITY”);

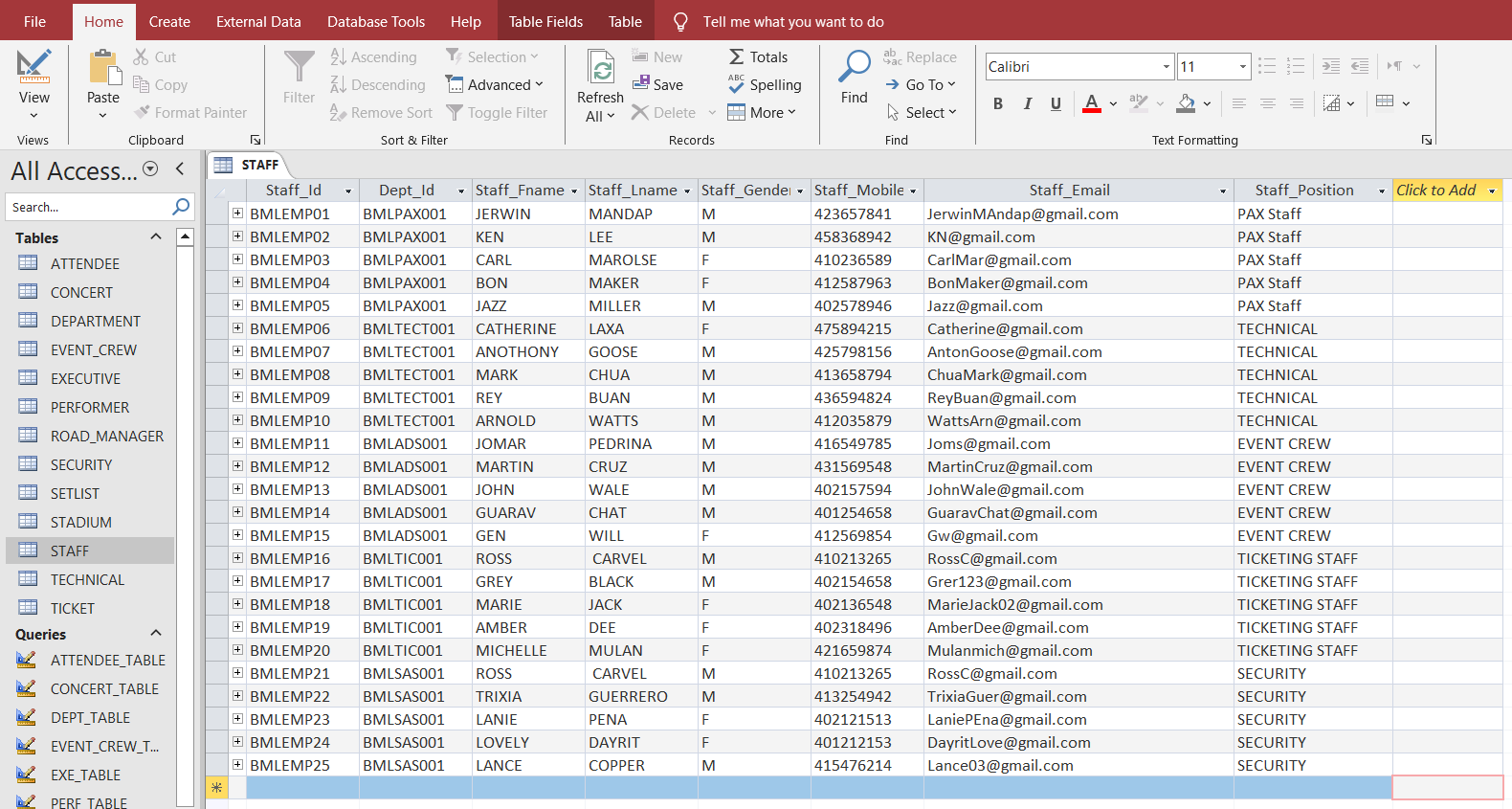


Figure ‑ Staff Records

* + - * 1. Insert Security Records

INSERT INTO SECURITY(Staff\_Id, Security\_License, Police\_Check)

VALUES

("BMLEMP21","2-145-458-368","1378130589"),

("BMLEMP22","2-001-225-898","1378130409"),

("BMLEMP23","2-124-569-789","1378130469"),

("BMLEMP24","2-896-895-748","1378130365"),

("BMLEMP25","2-694-758-654","1378130897");

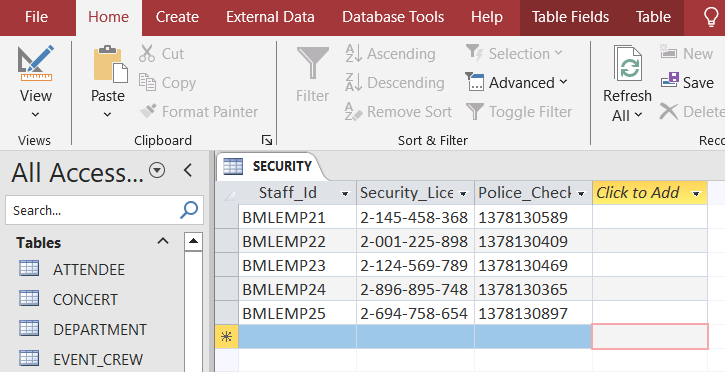


Figure ‑ Security Records

* + - * 1. Insert Technical Records

INSERT INTO TECHNICAL (Staff\_Id,Technical\_Certification, Years\_of\_Experience)

VALUES

("BMLEMP06","TSC546878","8"),

("BMLEMP07","TSC457896","9"),

("BMLEMP08","TSC321487","6"),

("BMLEMP09","TSC235874","4"),

("BMLEMP10","TSC452136","7");

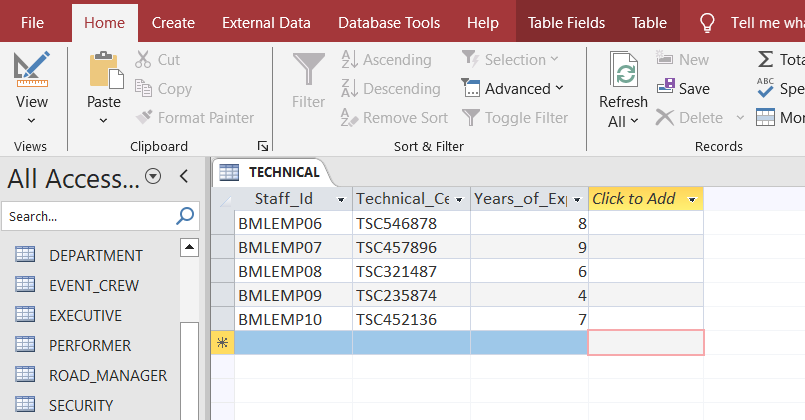


Figure ‑ Technical Records

* 1. Insert Event\_Crew Records

INSERT INTO EVENT\_CREW(Staff\_Id, Area\_Designation, Expertise)

VALUES

("BMLEMP11","VIP","Excellent in Verbal Communication"),

("BMLEMP12","Middle and upper seats","Proficient Organizing"),

("BMLEMP13","Stage","Designing"),

("BMLEMP14","Hallway and Balcony","Physical stamina"),

("BMLEMP15","Entrance and exit","Multitasking ability");

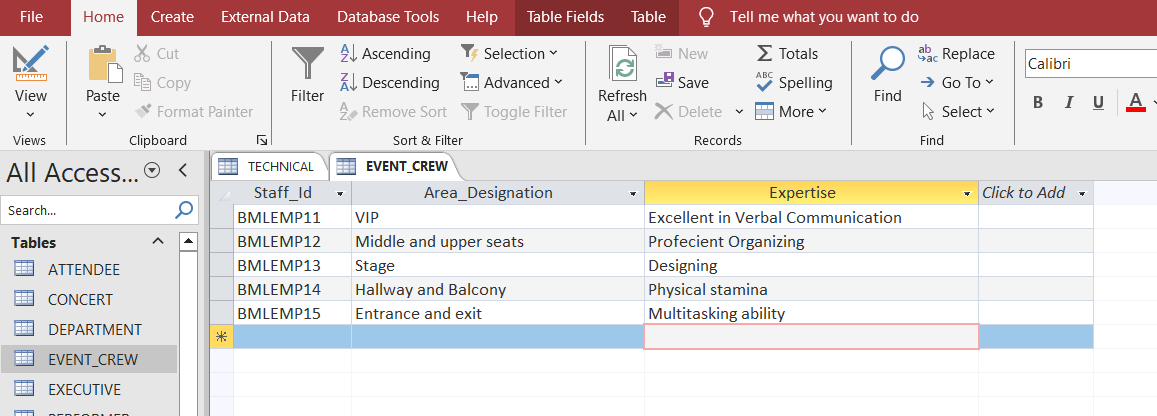


Figure ‑ Event\_Crew Records

# SQL QUERIES

In this chapter, sample queries are identified in which four of the queries are based on join queries. Kline et al (2024) states that a query is a formalized request for retrieving specific data or information from one or more database tables. This data may be created as SQL query results.

## 9.1. SELECT query with ORDER BY clause

This query retrieves all records from table Attendee and display them by ascending order in the column Attendee\_FName.

**SELECT \***

**FROM ATTENDEE**

**ORDER BY Attendee\_FName;**



Figure ‑ Query1: Select-Order by

## 9.2. SELECT query with WHERE clause

This query retrieves records from TICKET where the Stadium\_Id is equal to BMLSYD811 and Ticket\_Price is less than 280.

**SELECT \* FROM TICKET**

**WHERE Stadium\_Id = 'BMLSYD811' AND Ticket\_Price < 280;**

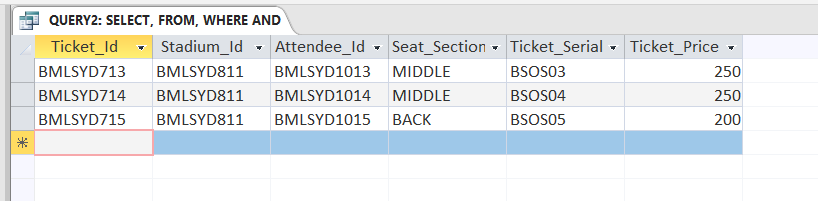
**

Figure ‑ Query2: Select-Where

## 9.3. INNER JOIN Query

This query retrieves all attributes and their attributes. An inner join is a join operation where Access only incorporates data from a table if there is matching data in the connected table, and vice versa. The attributes are stored in the PERFORMERS, and the attributes are stored in the SETLIST.

**SELECT Perf\_Fname, Perf\_Lname, Setlist\_Id, Song\_Title**

**FROM PERFORMERS**

**INNER JOIN SETLIST ON PERFORMERS.Performer\_Id = SETLIST.Performer\_Id;**

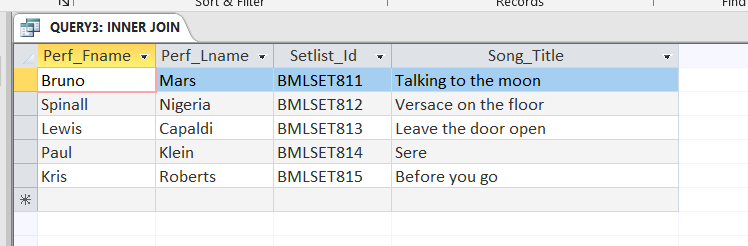


Figure ‑ Query3: Inner Join

## LEFT JOIN Query

This query retrieves all *table names* and their *attributes*. If a tablename has no *attribute*, the result is NULL on the order side. Left outer joins encompass all the entries from the first (left) table, regardless of whether there are corresponding values in the second (right) table.

**SELECT STAFF.Staff\_Fname, STAFF.Staff\_Lname, DEPARTMENT.Executive\_Id**

**FROM STAFF**

**LEFT JOIN DEPARTMENT ON STAFF.Dept\_Id = DEPARTMENT.Dept\_Id**

**ORDER BY DEPARTMENT.Executive\_Id;**

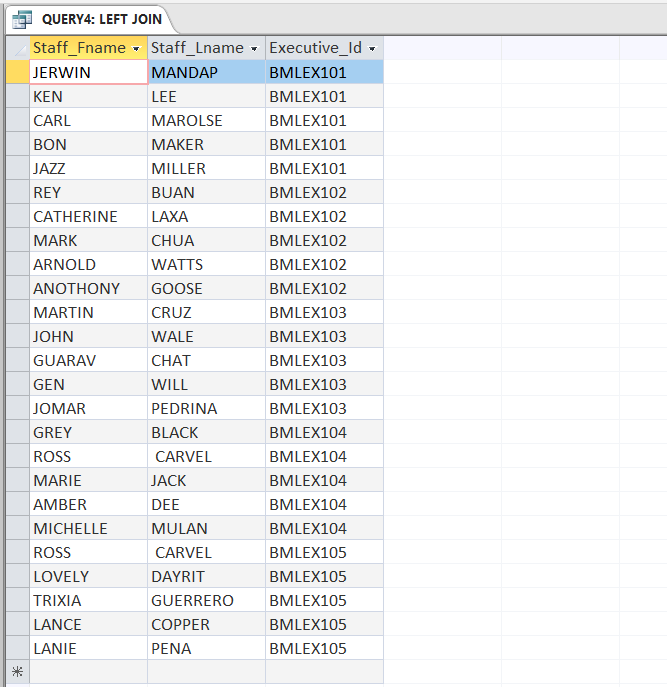
****

Figure ‑ Query4: Left Join

## RIGHT JOIN Query

Right outer joins get all entries from the second (right) table, regardless of whether there are matching values in the first (left) table.

**SELECT ROAD\_MANAGER.Rm\_Id, PERFORMER.Perf\_FName, PERFORMER.Perf\_LName, PERFORMER.Description**

**FROM ROAD\_MANAGER**

**RIGHT JOIN PERFORMER ON ROAD\_MANAGER.Rm\_Id = PERFORMER.Rm\_Id**

**ORDER BY ROAD\_MANAGER.Rm\_Id;**

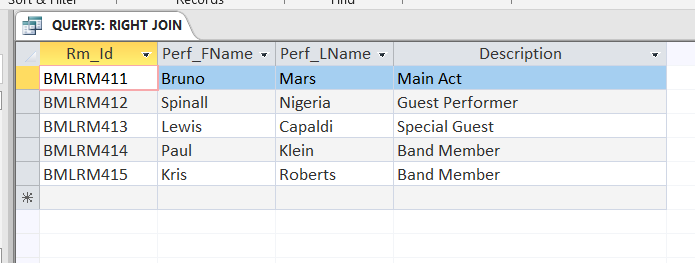
****

Figure ‑ Query5: Right Join

## FULL JOIN Query

A FULL OUTER join operation combines two record tables and includes all records from both sets, regardless of whether they satisfy the join criterion.

**SELECT ATTENDEE.Attendee\_FName, ATTENDEE.Attendee\_LName, TICKET.Ticket\_Id**

**FROM ATTENDEE**

**LEFT JOIN TICKET ON ATTENDEE.Attendee\_Id = TICKET.Attendee\_Id**

**UNION**

**SELECT ATTENDEE.Attendee\_FName, ATTENDEE.Attendee\_LName, TICKET.Ticket\_Id**

**FROM ATTENDEE**

**RIGHT JOIN TICKET ON ATTENDEE.Attendee\_Id = TICKET.Attendee\_Id**

**ORDER BY ATTENDEE.Attendee\_LName;**

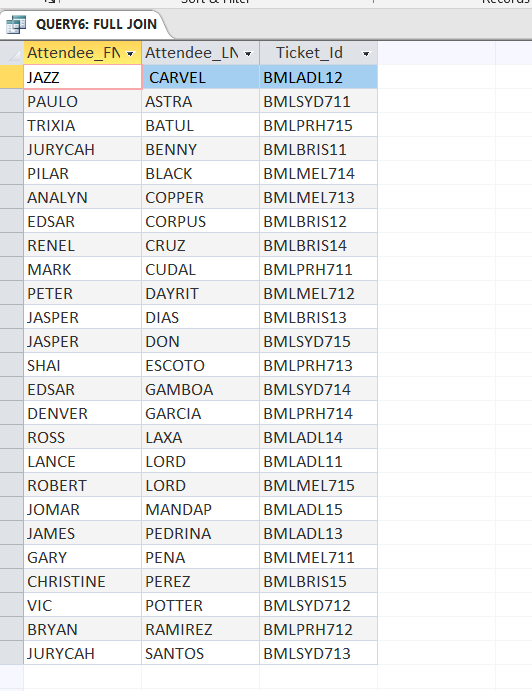
****

Figure ‑ Query6: Full Join

# REPORT

Reports allow users to show various elements of their database in a visually clear and printed layout. Access enables the creation of reports using both tables and queries.

## SELECT query with ORDER BY clause

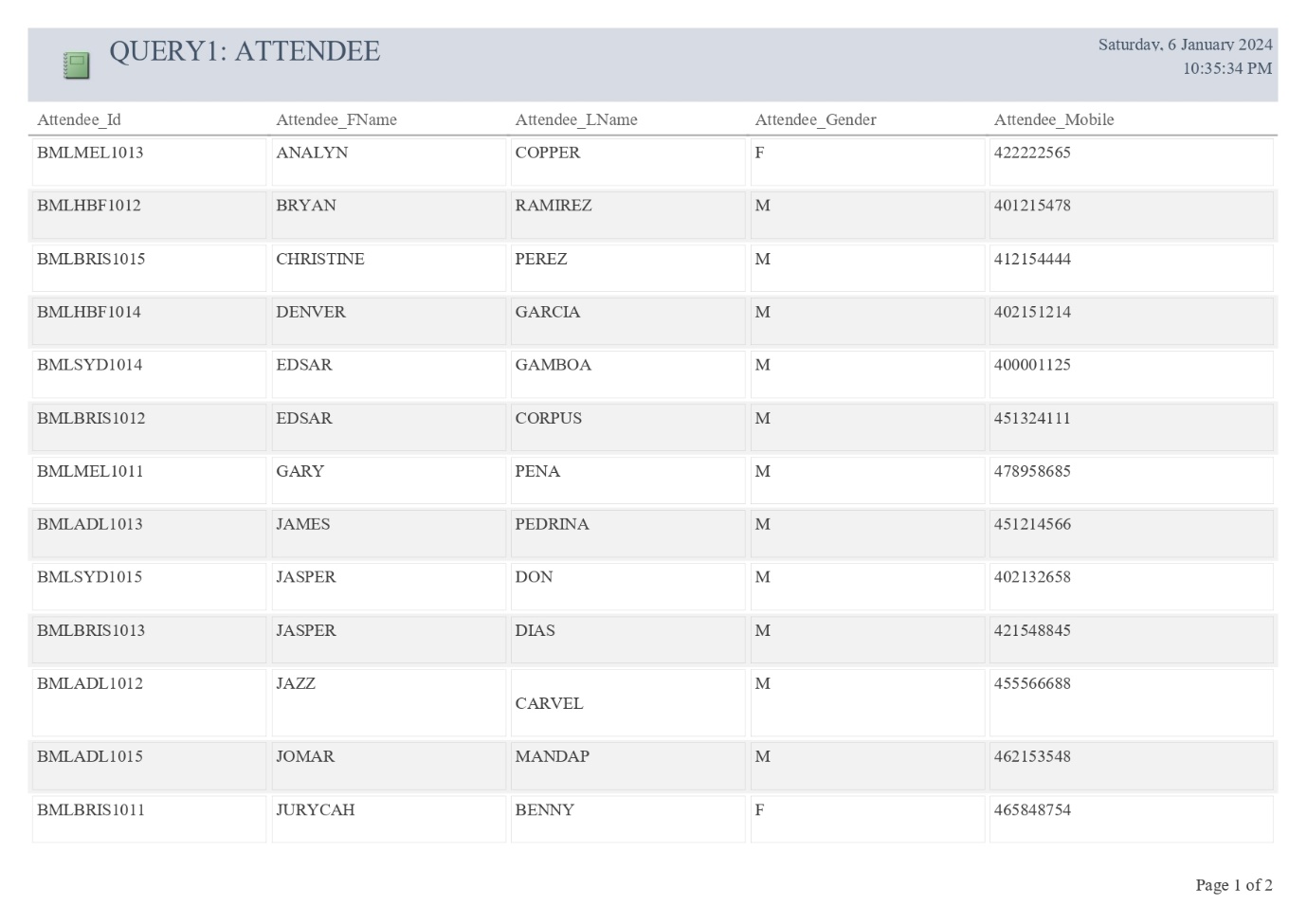


Figure ‑ Report1

## SELECT query with WHERE clause

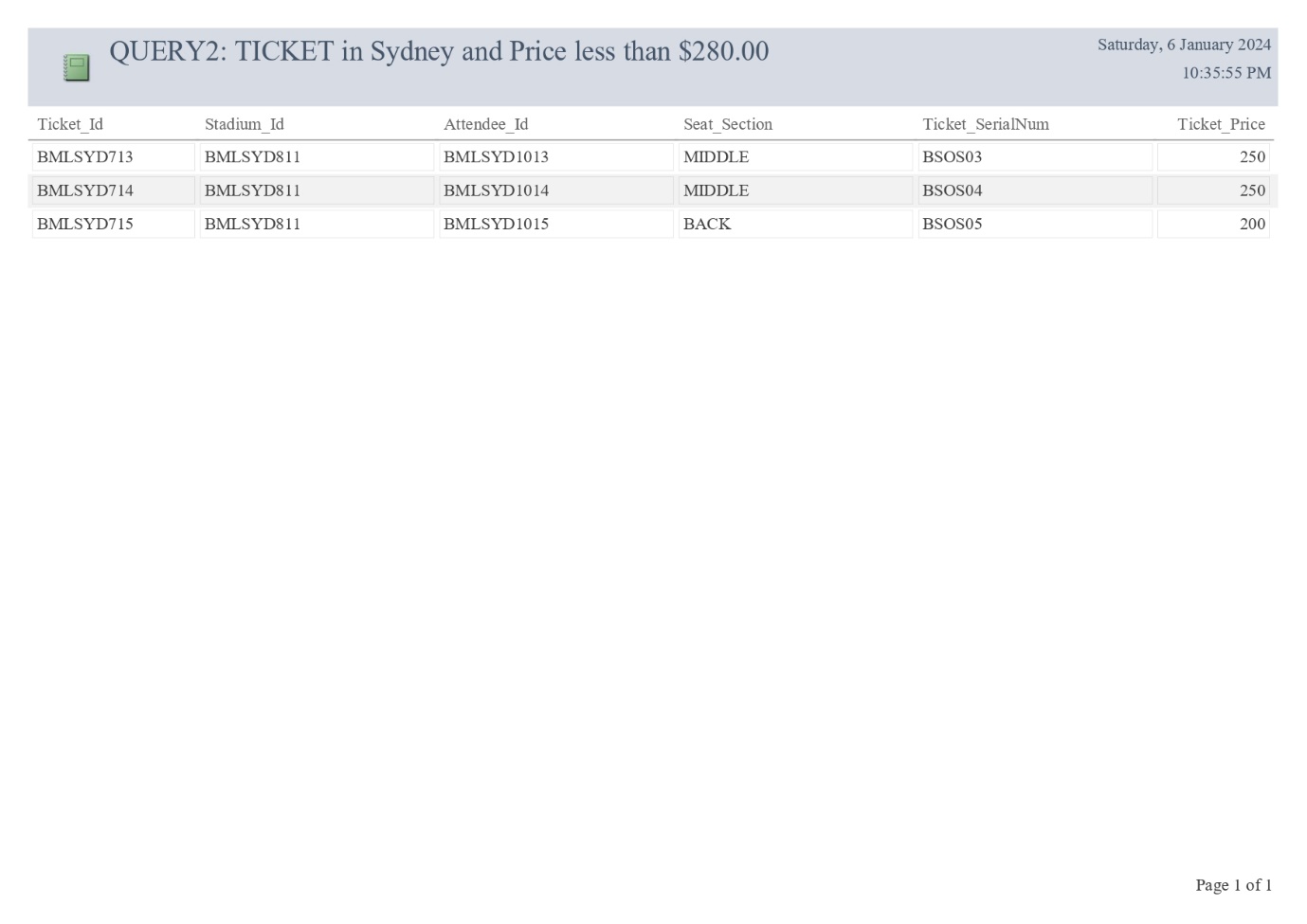


Figure ‑ Report2

## INNER JOIN Query



Figure ‑ Report3

## LEFT JOIN Query



Figure ‑ Report4

## RIGHT JOIN Query



Figure ‑ Report5

## FULL JOIN Query



Figure ‑ Report6

# REFERENCES

Halpin, T. and Morgan, T., 2010. Information modeling and relational databases. Morgan Kaufmann.

Kline, K.E., Kline, D. and Hunt, B., 2004. *SQL in a nutshell: a desktop quick reference*. " O'Reilly Media, Inc.".

Raipurkar, A. and Deokate, G., 2012. Business Rules in DBMS. *International Journal of Advanced Research in Computer Science*, *3*(3).

Rolik, O., Amons, O., Ulianytska, K. and Kolesnik, V., 2021. Modernization of the Second Normal Form and Boyce-Codd Normal Form for Relational Theory. In Advances in Computer Science for Engineering and Education III 3 (pp. 296-305). Springer International Publishing.

Teorey, L. J. (1990). Database Modeling and Design: Logical Design.

Teg Dainty. (2022). Bruno Mars Australia 2022. [Webpage]. Retrieved December 25, 2023, from https://www.tegdainty.com/tour/bruno-mars/