2024

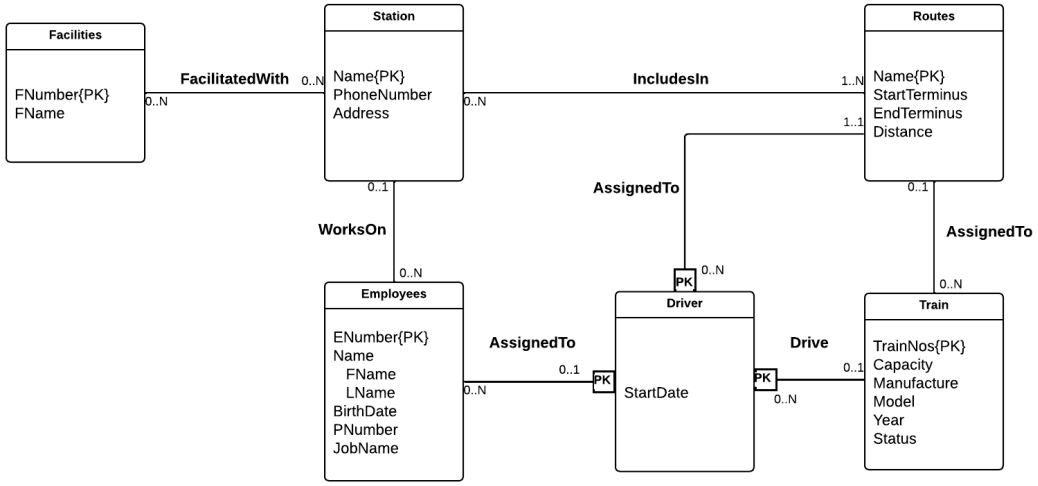
Database Concepts

ASSIGNMENT 1 DRAFT

MARTHIN DENDY LUMBAN TORUAN (S4075803)

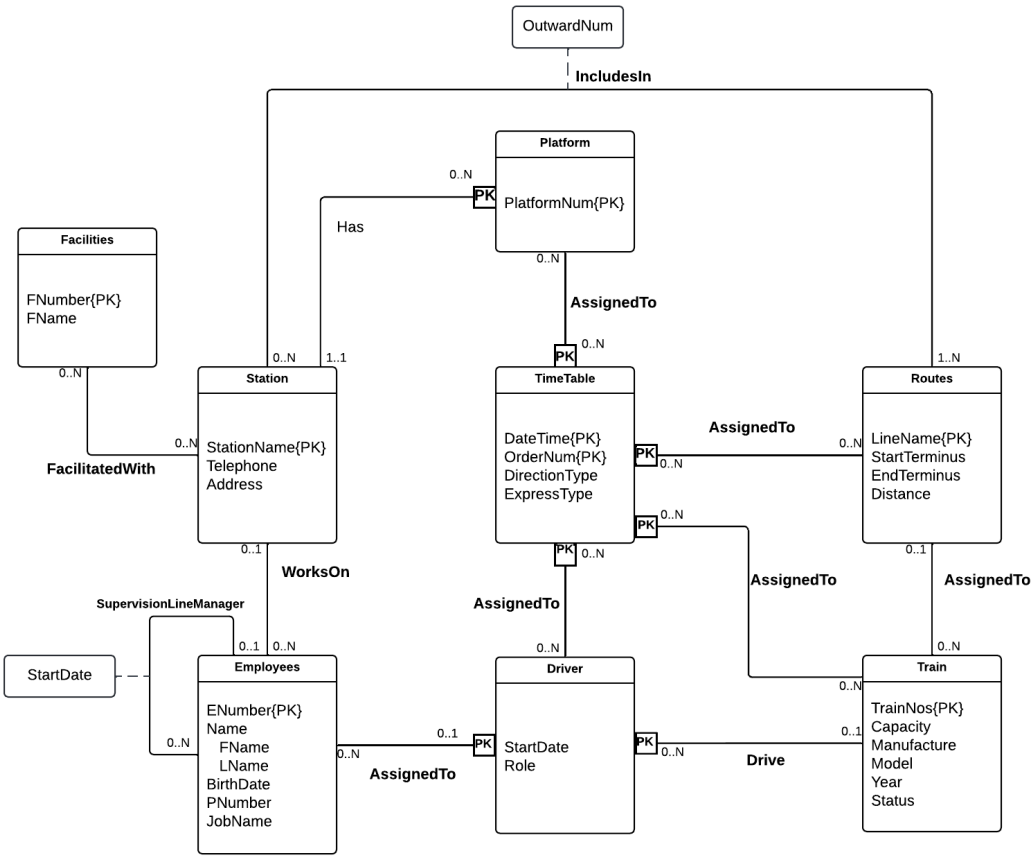
Task 1

ER Diagram Metro Trains



This ER diagram build based on limited information. There are five strong entities, which is have their own unique ID, such as **Station, Routes, Train, Employees**, and **Facilities**. I make an assumption that each facilities have their own unique identifier so for example, if station have more than one vending machine, each vending machine have their own id. Weak entities is the **Driver**, which this entities must related with employees, Train Number and the Lines Name. in this ER diagram it is assuming each station have several jobs for employee and one of them is Supervisor for Premium Station and can be described at JobName for Employees entities. Last thing to mention, there is not enough information to create the schedule or time table which is important to represent the relation of station, train and lines.

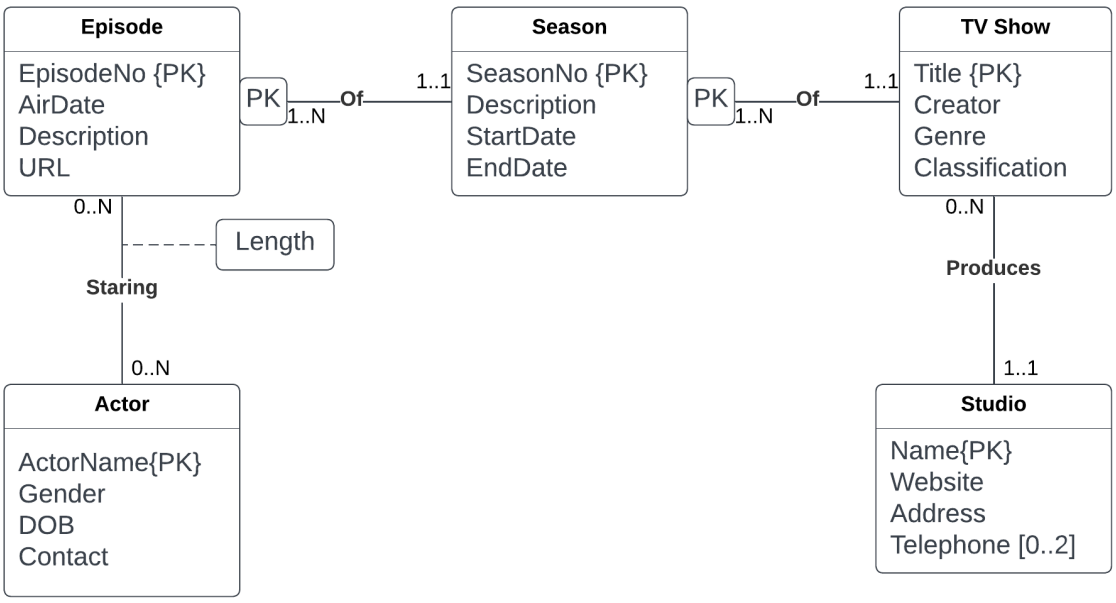
Task 2



This ER Diagram is refined from Task 1. New entities and relation added such as TimeTable, Platform, supervisor line manager. The assumption that I made is station connected to timetable through platform so there is no need to draw the relationship between timetables and station. Platform entities have composite key PlatformNumber and StationName. For Timetables there are attributes such as DateTime (specific date and time), DirectionType (Inbound or Outbound), and OrderNum is to indicate the inbound or outbound number. Then for ExpressType to indicate the type of trains will stop every station or not. With this relation the TimeTable could explain the relation which train number at specific time arrive at specific station and platform.

For driver there is added new attribute Role to indicate the primary driver or assistant driver. For relation between the station and Routes there is new attribute OutwardNum to indicate the number of station counted from central to end of lines or routes.

Task 3



Step 1. Map Strong Entities

Actor(ActorName,Gender,DOB,Contact)
TVShow(Title,Creator,Genre,Classification)
Studio(Name,Website,Address,Telephone)

Step 2. Map Weak Entities

Actor(ActorName,Gender,DOB,Contact)
TVShow(Title,Creator,Genre,Classification)
Studio(Name,Website,Address,Telephone)
Season(SeasonNo,Description,StartDate,EndDate)
Episode(EpisodeNo,AirDate,Description,URL)

Step 3. Map 1:1 Relationships

There are no 1:1 Relationships in this ER diagram. So this step is skipped

Step 4. Map 1:N Relationships

There are 3 1:N relationships:

1. Studio produce Tv show
2. Season of Tv show
3. Episode of Tv show

Actor(ActorName,Gender,DOB,Contact)
TVShow(Title,Creator,Genre,Classification,StudioName*)
Studio(Name,Website,Address,Telephone)

```
Season(TVShowTitle*,SeasonNo,Description,StartDate,EndDate)
Episode(TVShowTitle*,SeasonNo*,EpisodeNo,AirDate,Description,U
RL)
```

Step 5. Map M:N Relationships

Many to many relationships is starring relationships between actors and episode with attribute of length. So it must create new relationships named Starring.

```
Actor(ActorName,Gender,DOB,Contact)
TVShow(Title,Creator,Genre,Classification,StudioName*)
Studio(Name,Website,Address,Telephone)
Season(TVShowTitle*,SeasonNo,Description,StartDate,EndDate)
Episode(TVShowTitle*,SeasonNo*,EpisodeNo,AirDate,Description,U
RL)
Starring(TVShowTitle*,SeasonNo*,EpisodeNo*,ActorName*,Length)
```

Step 6. Map Multi-valued Attributes

The multi-valued attributes in this schema is Telephone number of studio, assuming the studio can have more than one phone number, such as mobile phone, fax, office phone. In this case I assume it needs to add another attribute to describe the phone number details.

```
Actor(ActorName,Gender,DOB,Contact)
TVShow(Title,Creator,Genre,Classification,StudioName*)
Studio(Name,Website,Address)
StdTelephone(StudioName*,TelephoneNum,Description)
Season(TVShowTitle*,SeasonNo,Description,StartDate,EndDate)
Episode(TVShowTitle*,SeasonNo*,EpisodeNo,AirDate,Description,U
RL)
Starring(TVShowTitle*,SeasonNo*,EpisodeNo*,ActorName*,Length)
```

Step 7. Map Higher-degree relationships

I assume Starring schema can be concluded as higher-degree relationships because it relates to other entities. So the schema is still same with step 6.

Optimization

I assume there are several things that need to do to make the database schema more efficient. For example there can be actor with the same name so I think it needs to make the DOB and ActorName became Primary composite key.

```
Actor(ActorName,DOB,Gender,Contact)
```

And because it is mandatory for the TVShow have StudioName so StudioName became Primary composite key with the Title. And to avoid if there are any same title but produced from different studio.

TVShow(StudioName*,Title,Creator,Genre,Classification)

Final Relational Database Schema

Actor(ActorName,DOB,Gender,Contact)
TVShow(StudioName*,Title,Creator,Genre,Classification)
Studio(Name,Website,Address)
StdTelephone(StudioName*,TelephoneNum,Description)
Season(TVShowTitle*,SeasonNo,Description,StartDate,EndDate)
Episode(TVShowTitle*,SeasonNo*,EpisodeNo,AirDate,Description,URL)
Starring(TVShowTitle*,SeasonNo*,EpisodeNo*,ActorName*,Length)

Task 4

Question 4.1

Foreign Key		Primary Key
Locations.country_id	----->	Countries.country_id
Departments.location_id	----->	Locations.location_id
Departments.manager_id	----->	Employees.employee_id
JobHistory.employee_id	----->	Employees.employee_id
Employees.empjob_id	----->	Jobs.job_id
Employees.department_id	----->	Departments.department_id
JobHistory.job_id	----->	Jobs.job_id
JobHistory.department_id	----->	Departments.department_id

Updated database schema reflecting **all** the constraints:
Employees(employee_id, first_name, last_name, phone_number, hire_date, empjob_id*, salary, department_id*)
Departments(department_id, department_name, manager_id*, location_id*)
Jobs(job_id, job_title, min_salary, max_salary)
Locations(location_id, street_address, postal_code, city, state_province, country_id*)
Countries(country_id, country_name)
JobHistory(employee_id*, start_date, end_date, job_id*, department_id*)

Question 4.2

No, Every employee can only have one job at the same period because each employee can only have maximum one Employees.empjob_id that related to Jobs.job_id.

Question 4.3

Yes, this schema allow an employee to work for different department in different period of time. Because in JobHistory the primary key is employee_id*, start_date, end_date so this only constrain one employee to finish their job at department at certain period of time.

Question 4.4

Technical yes, Because this department schema primary key is only department_id. So it allow to add new department_id with the same name but different location this will lead to confusing when identify the department. The best solution is to make new schema that represents the Office brach which relates to department schema.

Question 4.5

This query will work if the employee with salary 66000 is only Adam Smith. But this query is not best practice because to change the attribute value for Adam Smith should refer to the unique id of the employee that is employee_id which is 50. So, The query should be:

```
UPDATE Employees SET empjob_id=33, hire_date='2012-10-02' WHERE
employee_id=50;
```

If this query executed, then the request “find all the past contracts that Adam Smith used to have” is impossible because it is already replaced with new information. So to fulfill this request the JobHistory must add new record according to Employee attribute value of Adam Smith which is:

```
INSERT INTO JobHistory VALUES(50, ’2009-10-26’, ’2012-10-02’, 22, 2);
```

This INSERT INTO JobHistory query must executed before running the UPDATE Employees query. So it is possible to find Adam Smith pas contracts.

Question 4.6

This DELETE query will result error because the location_id is foreign key to table Departments attribute. So this violating Referential Integrity Constraint. So if the office want to move other place, it must insert new office location first in the Locations table. Then update the location in Departments. Or it could update the location of the department into NULL until there are new Location. After that the old location can be deleted if it is not used at all.

Question 4.7

```
CREATE TABLE JobHistory(
employee_id INTEGER NOT NULL,
start_date TEXT,
end_date TEXT,
job_id INTEGER,
department_id INTEGER,
PRIMARY KEY(employee_id, start_date, end_date),
FOREIGN KEY(employee_id) REFERENCES Employees(employee_id),
FOREIGN KEY(job_id) REFERENCES Jobs(job_id),
FOREIGN KEY(department_id) REFERENCES Departments(department_id),
);
```

Question 4.8

This query will result error because violation of Referential integrity constrain where the foreign key of department_id 4 is not exist when assign the employee. So the solution is to crate the department first with NULL manager_id. The query should be:

```
INSERT INTO Departments VALUES (4, ‘Art’, NULL, 20);

INSERT INTO Employees VALUES (88, ‘Scott’, ‘Wallace’, ‘1111’, ‘2020-01-01’,
10, 140000, 4);
UPDATE Departments SET VALUES manager_id=88 WHERE department_id=4;
```