1923636

Ellaine Fontamillas

C6-QMD-11

Querying and Managing Databases

End Assessment

Faculty of Computing

QMD Project Report

1. **A. Project Description**

This project implements the bus fleet management of Lesotho in a system. It consists of entities and attributes surrounding the bus transportation, its passengers and parcels that move district to district.

**B. System Requirements**

- Automated Passenger ID

- Proper Primary Key identifications

- Appropriate Foreign keys for linking

- Creation of entities and their records

- A normalized table of records

1. **Normalization**

**Bus Fleet table** = {**PK:** Bus Fleet Code, Bus Fleet Name, No. of buses}

|  |  |  |
| --- | --- | --- |
| Bus Fleet | | |
| Bus Fleet Code | **Bus Fleet Name** | **No. of Buses** |
| AZ123 | Ha-Mafafa | 10 |
| AY124 | Sea Point | 11 |
| AX125 | Mazenod | 12 |
| AW126 | Tallman | 6 |
| AV127 | Luzon | 3 |
| AU128 | Leqele | 5 |
| AT129 | Thetsane | 5 |
| AS130 | Industrial | 4 |
| AR131 | Maine | 2 |
| AQ132 | Coon | 11 |

This table is normalized. Each Primary key is unique and all other attributes are non-prime and independent

**Bus table** = {**CK:** Bus Code = (Bus fleet code + 2 letters), Class, type, Arrival time, departure time, driver}

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Bus | |  | |  |  |  |  |  |  |
| Bus Fleet Code +  Bus Code | | | **Type** | **Class** | **Arrival Time** | **Departure Time** | **Duration** | **Source** | **Destination** |
| AZ123 | AR | | ELDERLY | ORDINARY | 1300 | 1000 | 3 hours | MASERU | QACHA’S NEK |
| AY124 | BR | | ELDERLY | PRESTIGE | 1300 | 1000 | 3 hours | BEREA | MOKHOTLONG |
| AX125 | CR | | ELDERLY | ORDINARY | 1200 | 1000 | 2 hours | BUTHA-BUTHE | MOHALE’S HOEK |
| AW126 | DR | | ELDERLY | REPUTABLE | 1100 | 1000 | 1 hour | LERIBE | MAFETENG |
| AV127 | ER | | ELDERLY | REPUTABLE | 1100 | 1000 | 1 hour | MAFETENG | LERIBE |
| AU128 | FR | | YOUTH | ORDINARY | 1300 | 1000 | 3 hours | MOHALE’S HOEK | BUTHA-BUTHE |
| AT129 | GR | | YOUTH | ORDINARY | 1300 | 1000 | 3 hours | MOKHOTLONG | BEREA |
| AS130 | HR | | YOUTH | PRESTIGE | 1100 | 1000 | 1 hour | QACHA’S NEK | MASERU |
| AR131 | IR | | YOUTH | ORDINARY | 1500 | 1000 | 5 hours | **QUTHING** |  |
| AQ132 | JR | | YOUTH | REPUTABLE | 2000 | 1800 | 2 hours | **THABA-TSEKA** |  |

This table is normalized. Each Primary key is unique as it is made up of two prime attributes – candidate keys, and all other attributes are non-prime and independent.

**Bus Ranks table**= {**PK:** Bus Rank Name, **FK:** Bus Code, No. of buses left, No. of buses arrived, **FK:** Working Employee – Secretary ID}

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Bus Rank | |  |  |  |
| Bus Name | **Bus Code** | **No. of Buses left** | **No. of Buses arrived** | **Working Employee – Secretary ID** |
| MASERU | AZ123AR | 30 | 22 | 3532414 |
| BEREA | AY124BR | 35 | 44 | 6764345 |
| BUTHA-BUTHE | AX125CR | 66 | 14 | 6564563 |
| LERIBE | AW126DR | 45 | 15 | 7845623 |
| MAFETENG | AV127ER | 44 | 20 | 7262563 |
| MOHALE’S HOEK | AU128FR | 22 | 30 | 6547322 |
| MOKHOTLONG | AT129GR | 19 | 54 | 2642626 |
| QACHA’S NEK | AS130HR | 23 | 66 | 6734656 |
| QUTHING | AR131IR | 45 | 60 | 6562345 |
| THABA-TSEKA | AQ132JR | 35 | 33 | 5674572 |

**Passengers table** = {**CK:** Passport No. + Passenger ID, Name, Age, Sex, Phone No, Address}

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Passengers | | | |  |  |  |  |  |  |  |  |
| Passenger ID  Passport ID | | **Bus Code** | **First Name** | | **Last Name** | **Phone No** | **District** | **Village** | **Street** | **Age** | **Sex** |
| 1 | **EC2876** | AZ123AR | HELEN | | GEORGE | 905562 | Maseru | Thetsane | 123 Acre. | 21 | F |
| 2 | **EC5677** | AY124BR | MAINE | | COON | 905661 | Maseru | Sea point | 2398 Ave. | 31 | M |
| 3 | **EC4675** | AX125CR | LEONARD | | DAWN | 905660 | Mafeteng | Maria | 12 Brows | 4 | M |
| 4 | **EC8974** | AW126DR | LEAH | | RUTH | 905663 | Leribe | Zion | 901 Stw. | 18 | F |
| 5 | **EC2373** | AV127ER | SAMANTHA | | NEWN | 905664 | Quthing | Maputsoe | 4325 Rd. | 5 | F |
| 6 | **EC0972** | AU128FR | CHEN | | LEE | 956655 | Berea | Ha leqele | 393 Strt. | 78 | M |
| 7 | **EC2371** | AT129GR | PACE | | PLOON | 905666 | Maseru | Thamae | 0101 Ave. | 60 | M |
| 8 | **EC1270** | AS130HR | ENRIQUE | | VELASQUEZ | 905667 | Quthing | West | 9302 Mel. | 57 | M |
| 9 | **EC7678** | AR131IR | RICKY | | DAWN | 905668 | Quthing | East | 893 Hdl. | 19 | M |
| 10 | **EC4479** | AQ132JR | CECELIA | | MANISTER | 905687 | Leribe | Pulo | 911 OJK. | 54 | F |

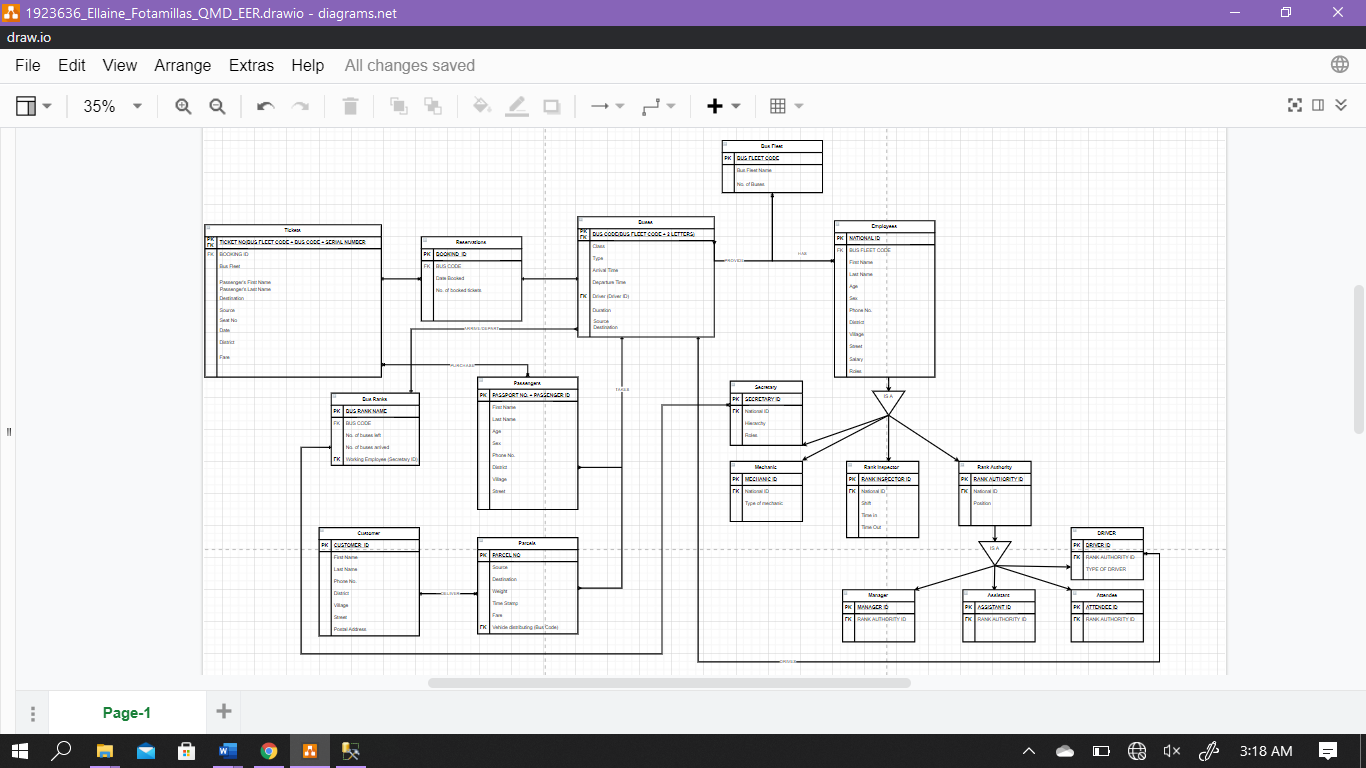
This table is normalized. Each non-prime attribute is independent, primary keys/candidate keys are unique, multivalued attributes like address has been simplifies, Name has been simplified as well.

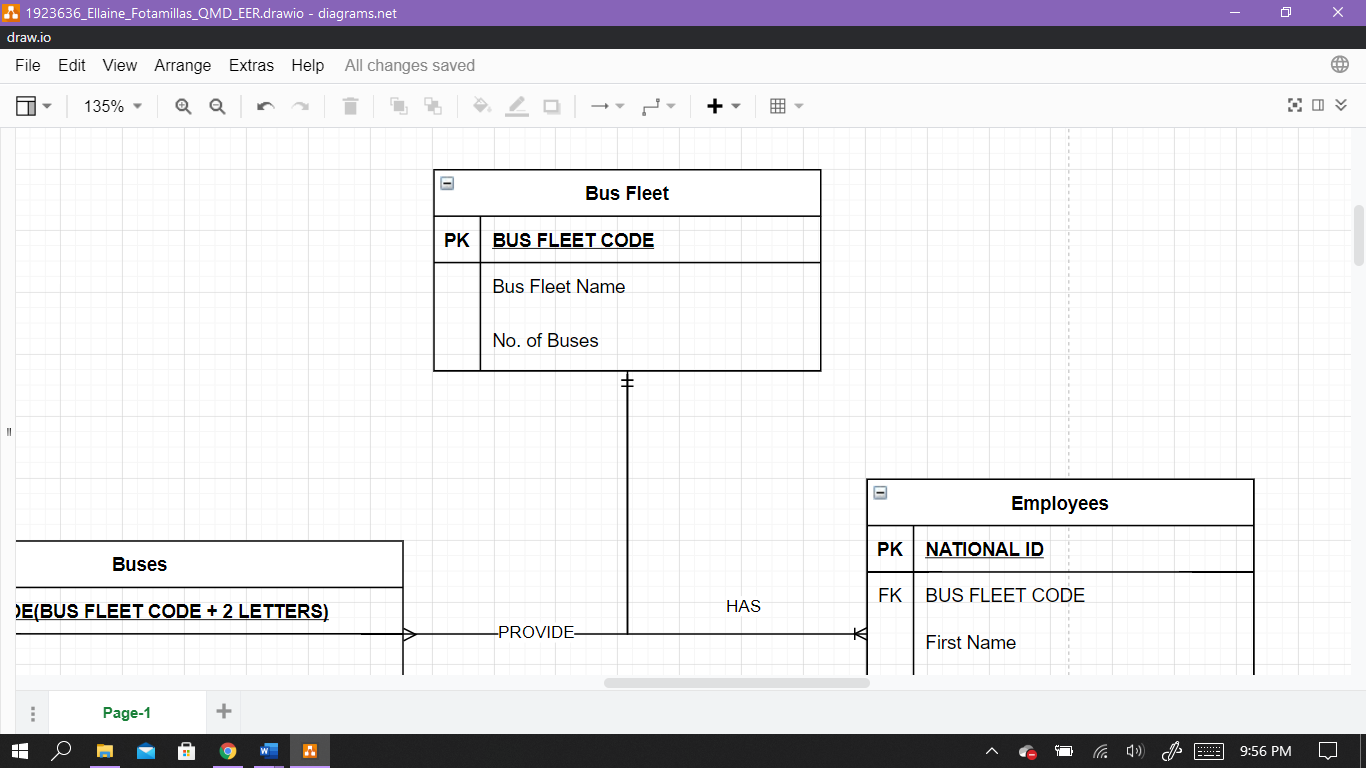
**Other tables:**

* **Tickets table** = {**CK:** Ticket No: (Bus Fleet Code + Bus Code + 4 Serial Numbers, **FK:** Booking ID, Bus Fleet Name, passenger’s first name, passenger’s last name, Source, Destination, Seat No, date, District, fare}
* **Reservations table** = {**PK:** Booking ID **FK:** Bus Code, Date Booked, No. of booked tickets}
* **Parcels table** = {**PK:** Parcel No, **FK:** Bus Code, Source, Destination, Weight, Time Stamp, Fare}
* **Customers** = {**PK:** Customer ID, First name, Last Name, Phone Number, District, Village, Street, Postal Address}
* **Employees table** = {**PK:** National ID, **FK:** Bus Fleet Code, First Name, Last Name, Age, Sex, Phone No, District, Village, Street, Salary, Roles}
* **Secretary table** = {**PK:** Secretary ID **FK:** National ID, Hierarchy, Roles}
* **Mechanic table** = {**PK:** Mechanic ID **FK:** National ID, Type of mechanic}
* **Rank Inspector table** = {**PK:** Rank Inspector ID **FK:** National ID, Shift, Time in, Time Out}
* **Rank Authority table** = {**PK:** Rank Authority ID **FK:** National ID, Position}
* **Manager table** = {**PK:** Manager ID **FK:** National ID}
* **Attendee table** = {**PK:** Assistant ID **FK:** National ID}
* **Assistant table** = {**PK:** Attendee ID **FK:** National ID}
* **Driver table** = {**PK:** Driver ID **FK:** National ID, Type of driver}

Similar to the ones with records in the tables, these other entities follow the normalization rules which they followed.

1. **Normalized relational schema**

**Overview**

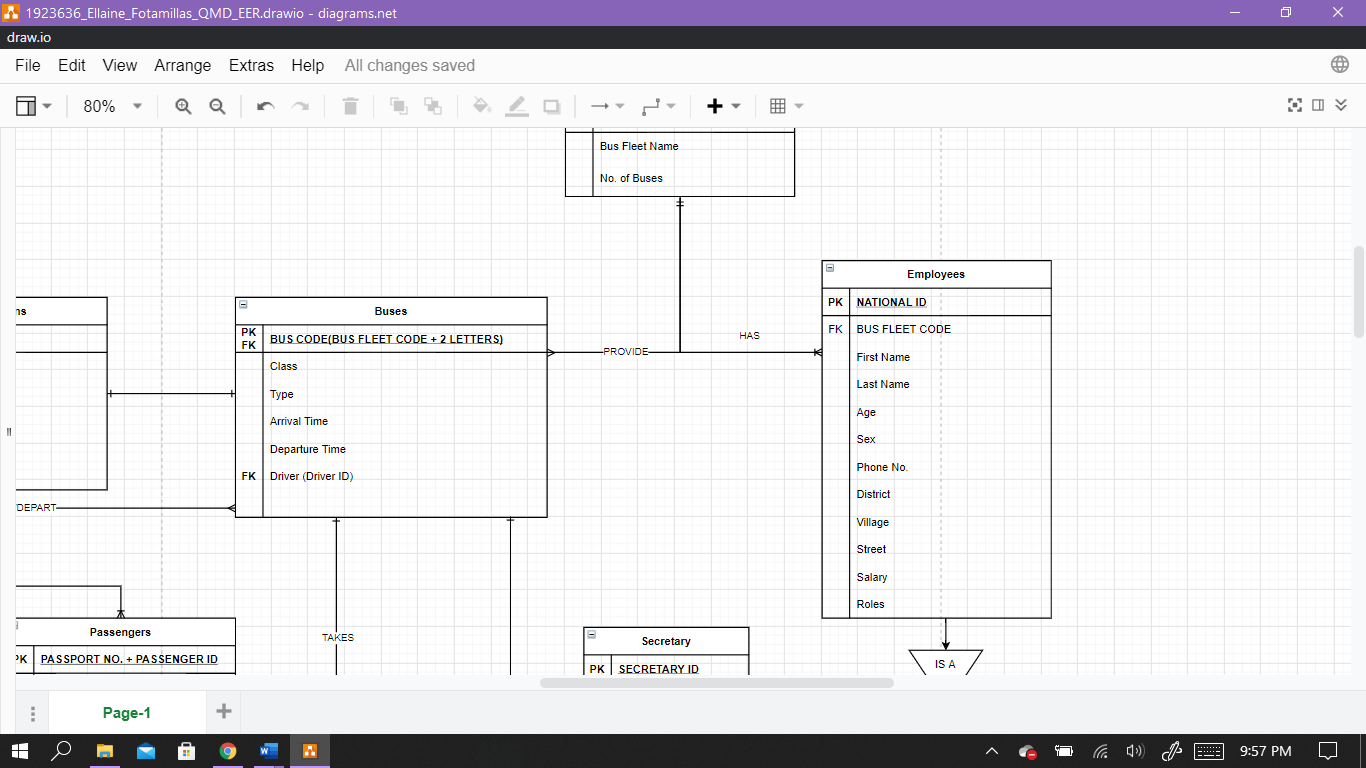


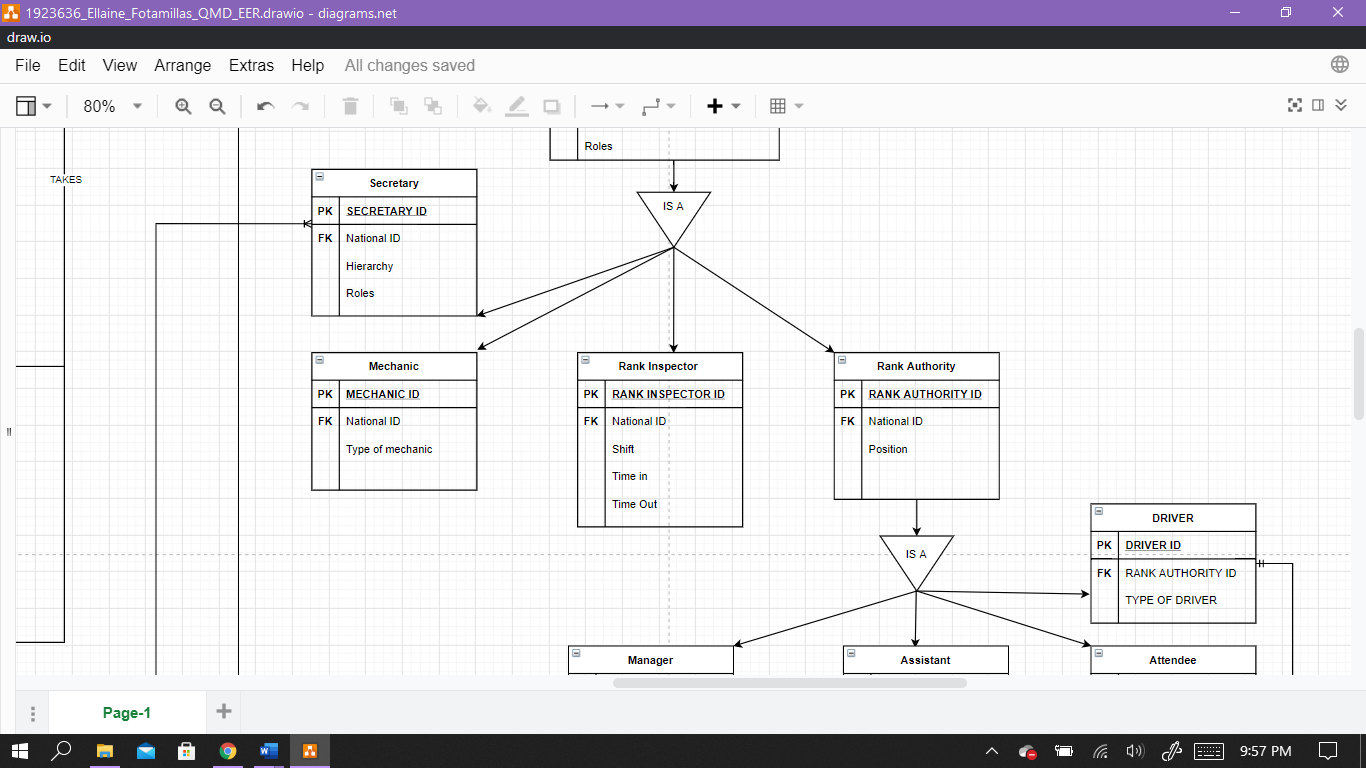
Bus Fleet Entity has a relationship between Employees and Buses.

* **A** Bus Fleet Provides **Buses**
* **A** Bus Fleet has **employees** working for it.

The Employee entity has a unique ID – National ID. It stores attributes about the employee from name to address. Address is normalized into single attributes.

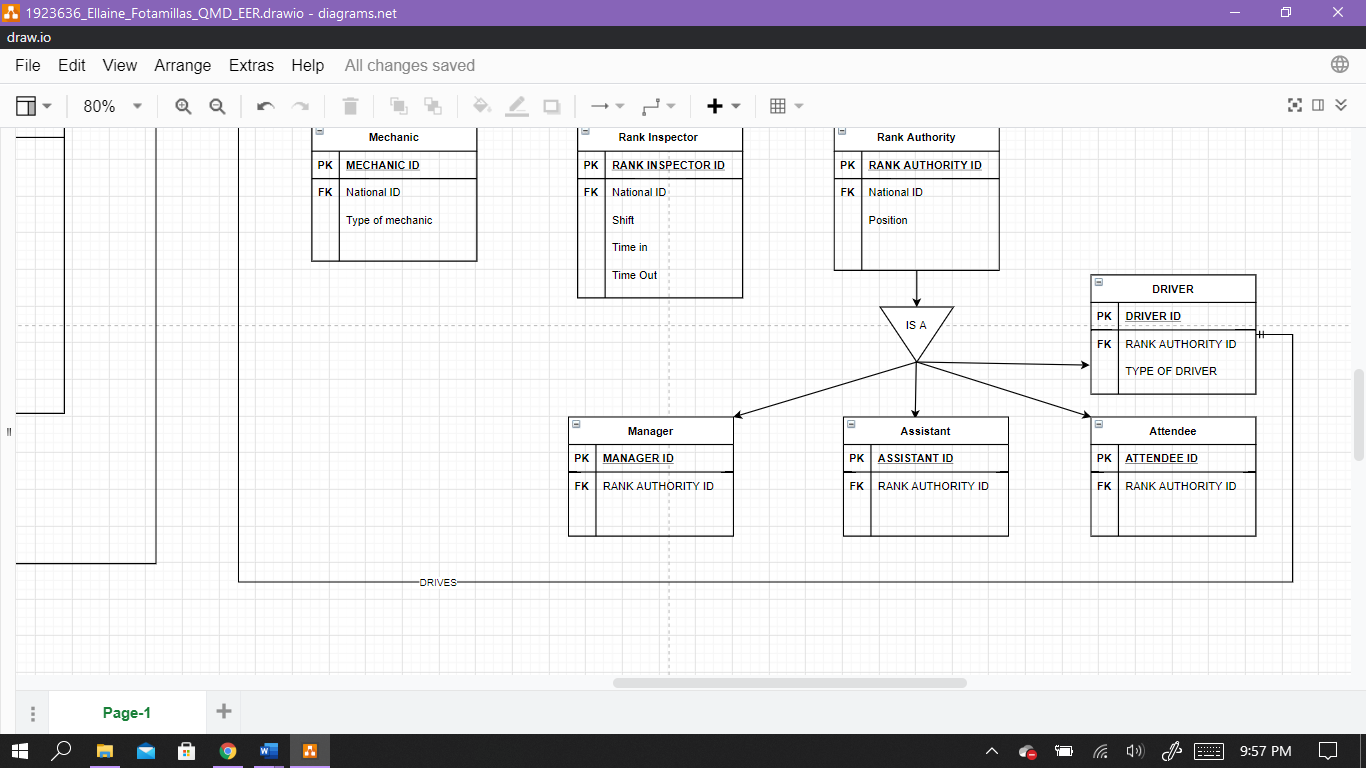
To link the bus fleet entity to employee entity we provide the bus fleet code as the foreign key, as an employee can work in a bus fleet, therefore having a bus fleet code

****

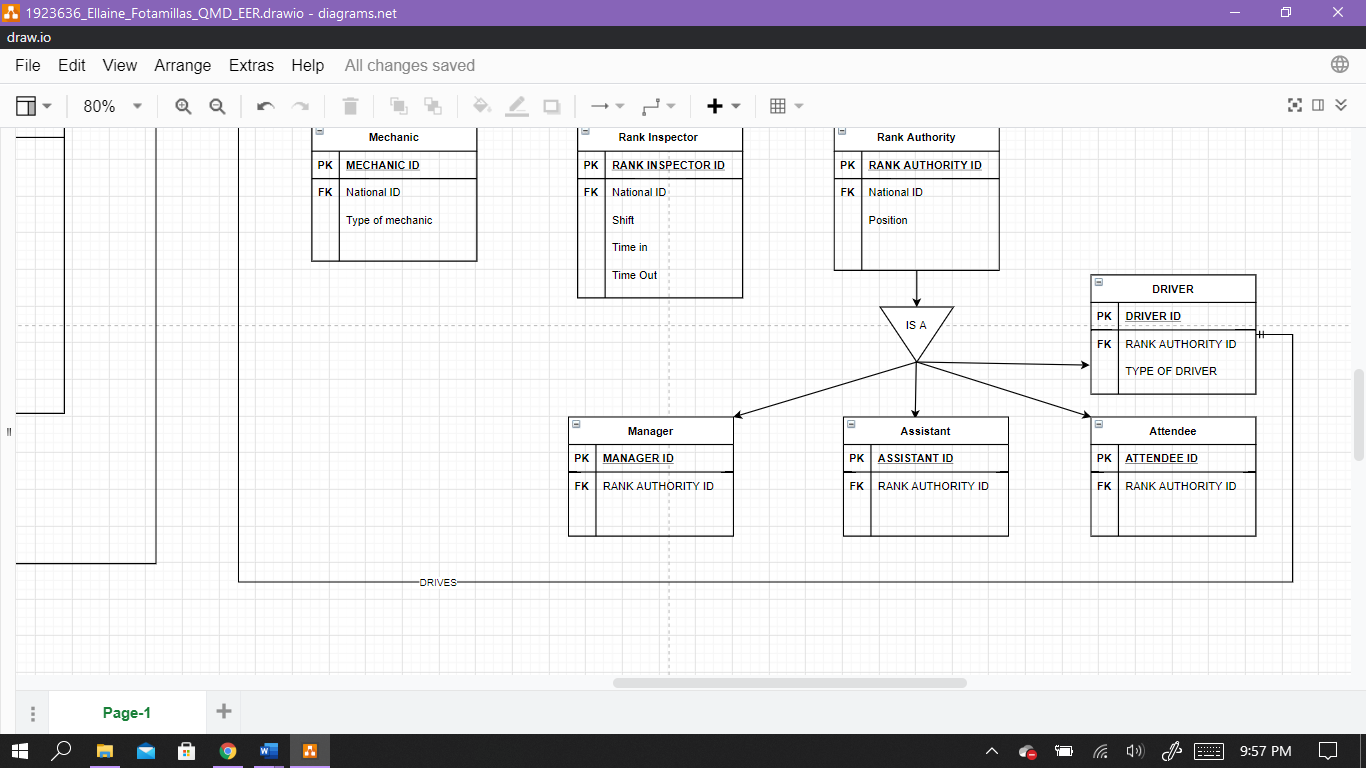
****

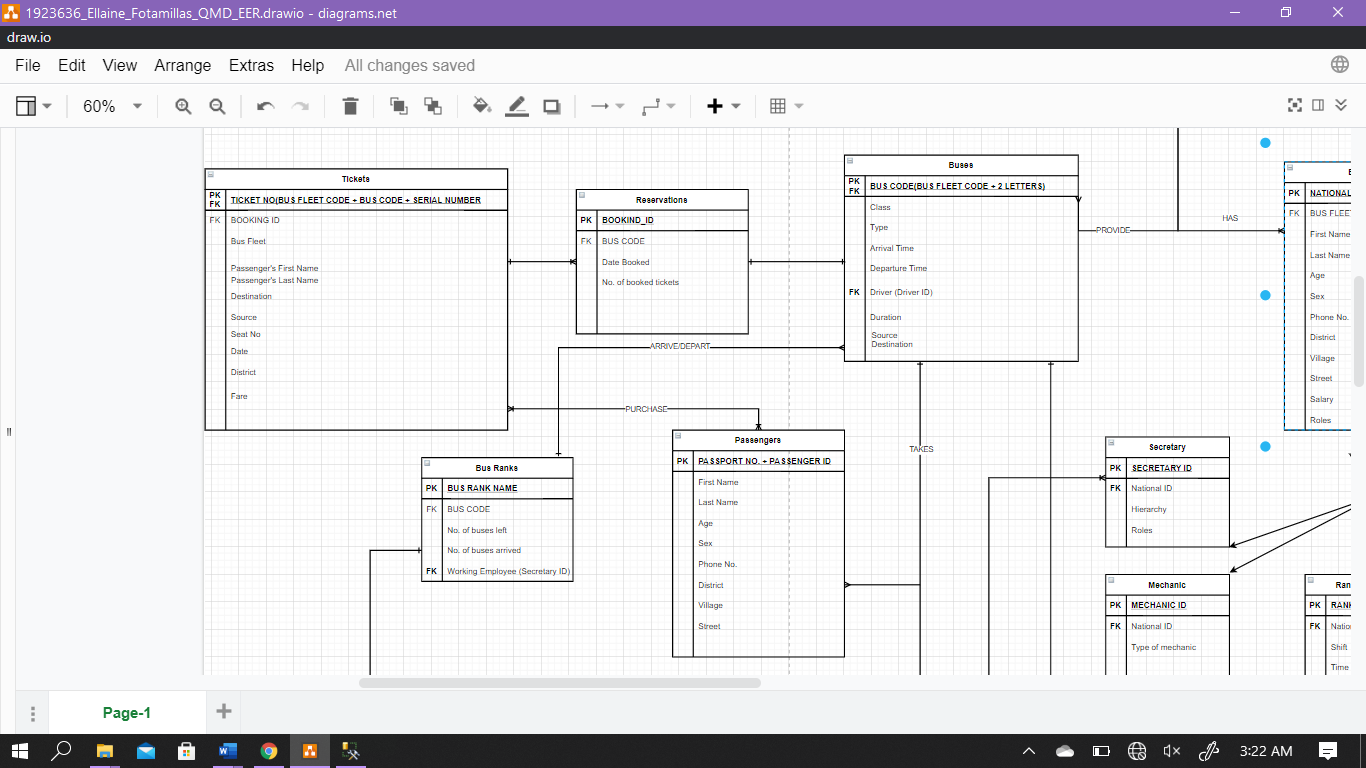
Again, the employees entity

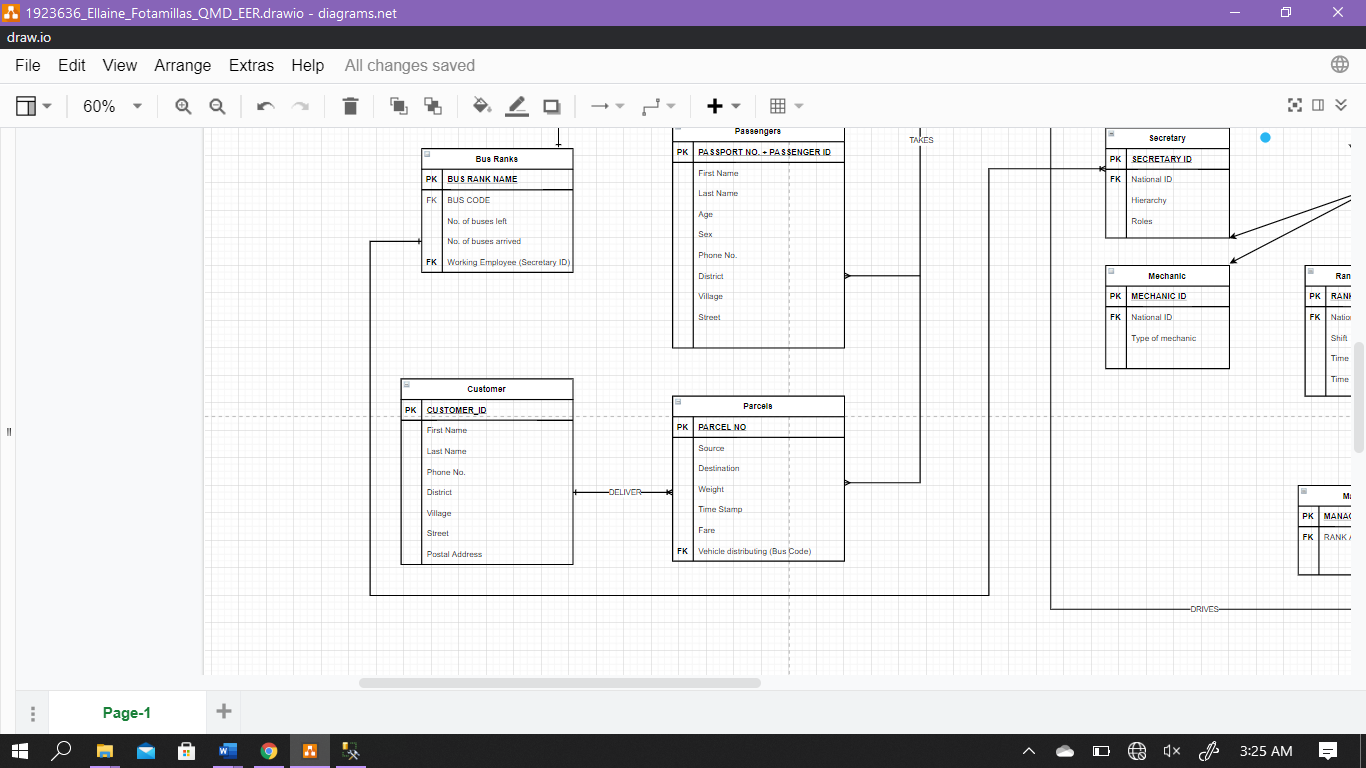
Again, the employees entity creates a supertype/subtype relationship. We can see a top-down approach as the employee can be specialized as either a secretary, mechanic, rank inspector or rank authority. Each subtype inherits from the supertype as they are still employees so they have details when being recorded. It also has its own unique attributes. Here the inherited Primary key from the employee table is inserted as a foreign key instead as there needs to be a link between the entities. A primary is created here for them as it is not possible to create the entity with a foreign key but no primary key.

****

Again, the rank authority subtype becomes a supertype – showing another top-bottom approach. This is because the authority can be either a manager, assistant, attendee or driver. Instead of these entities just becoming a record under the position attribute in rank authority, they are depicted as entities having their own unique attributes. They are similar to the subtypes of employees. Specifically, the Driver entity is needed to show its relationship with the other entities. To be more precise, driver ID is used to identity the **driver** that **drives** the **bus(es).**

****

****

****

On the left side –

The relation

with the bus fleet

and buses, we can

see that the bus

takes two entities:

Passengers and

Parcels

The relationship:

The **bus(es)** takes **passengers** and **parcels**

Because passengers and bus does not have a direct relationship, they do not share a key

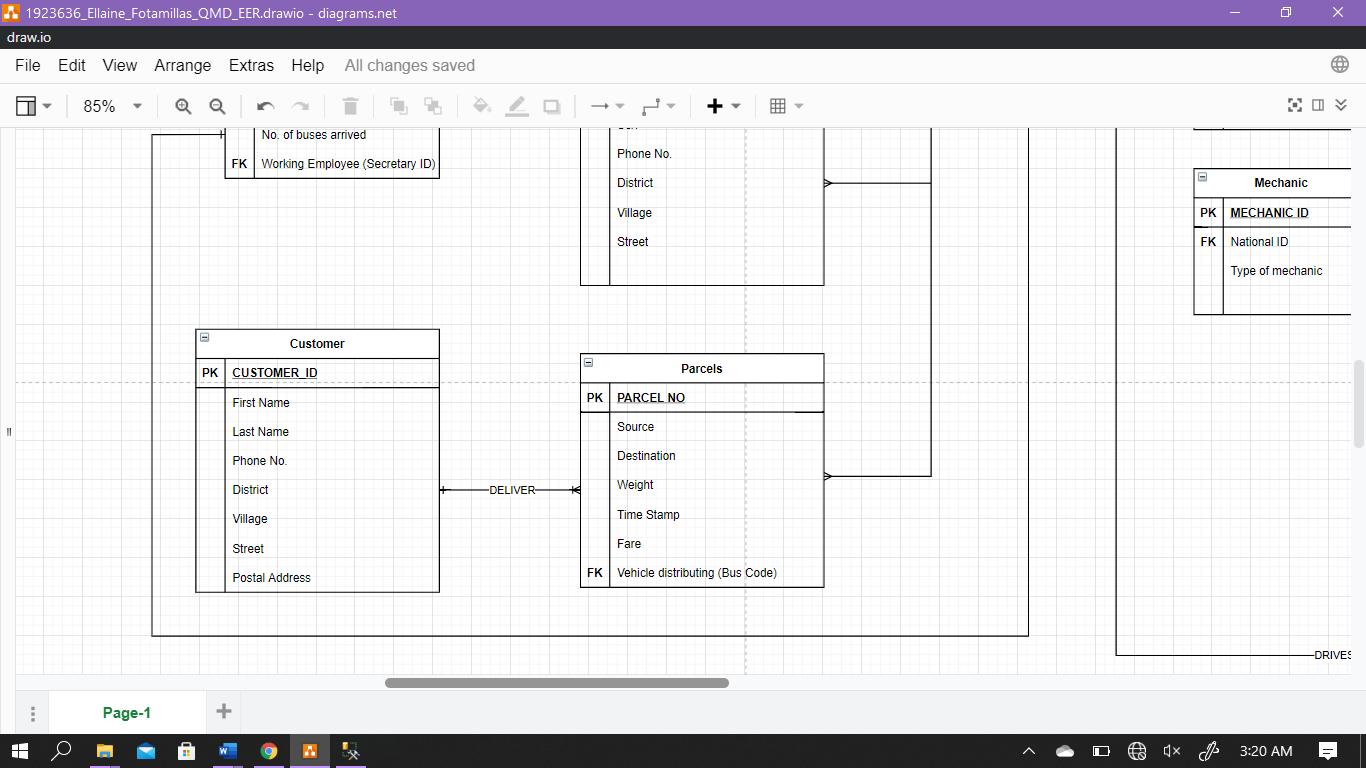
The **passenger(s) purchase tickets** which have **reservation(s)**

This reservation(s) is for bus seats in a **bus**

And the **bus arrives** or **departs** at a **bus rank**

A **bus ranks** has **secretaries** to run administrative duties thus the foreign key – Secretary ID

This is the relationship with the Driver entity

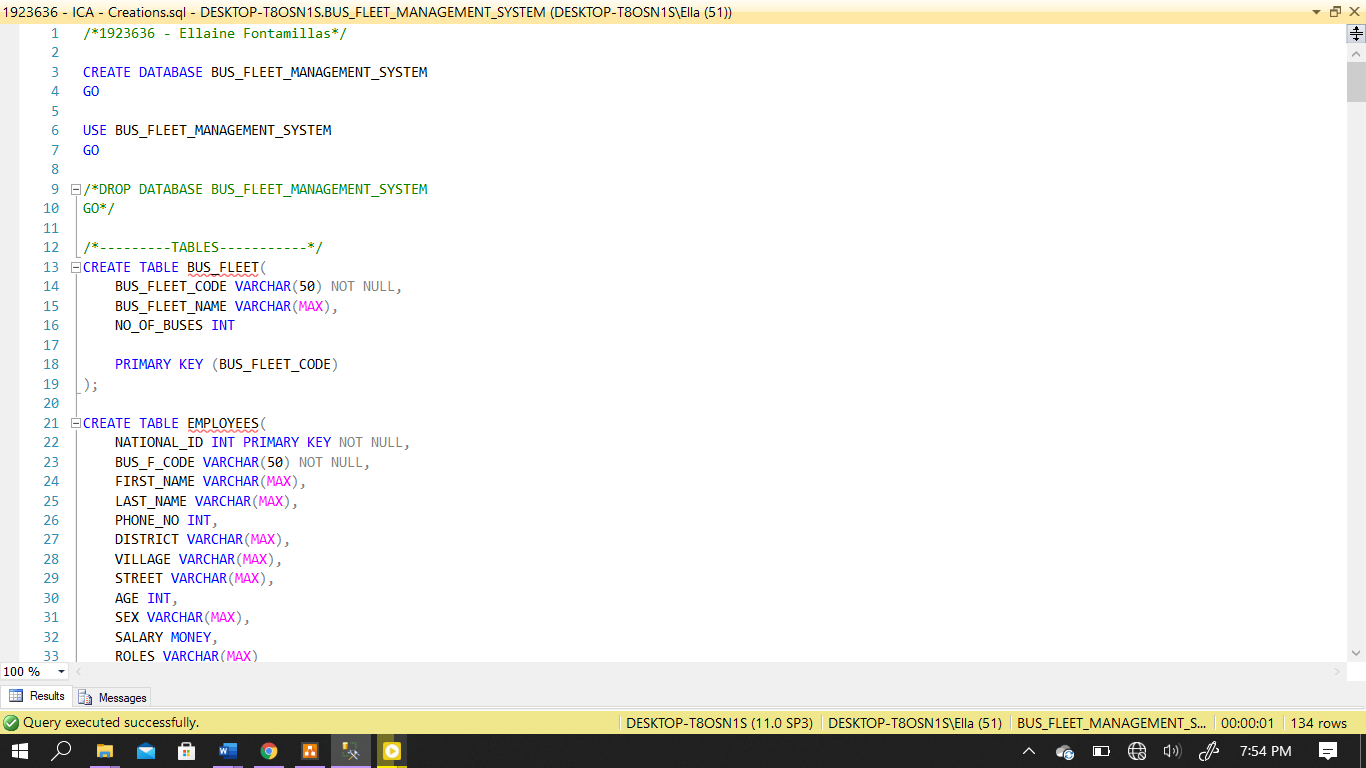
****

Here the relationship is:

**bus takes parcels** and **delivers** them to the **customer.**

**Parcels** has a foreign key that ultimately shows who distributed the parcel in case of the customer having an issue with them.

1. **SQL Queries**

****

* Lines of code: 1 to 63 -

This is the script for creating the databases and tables.

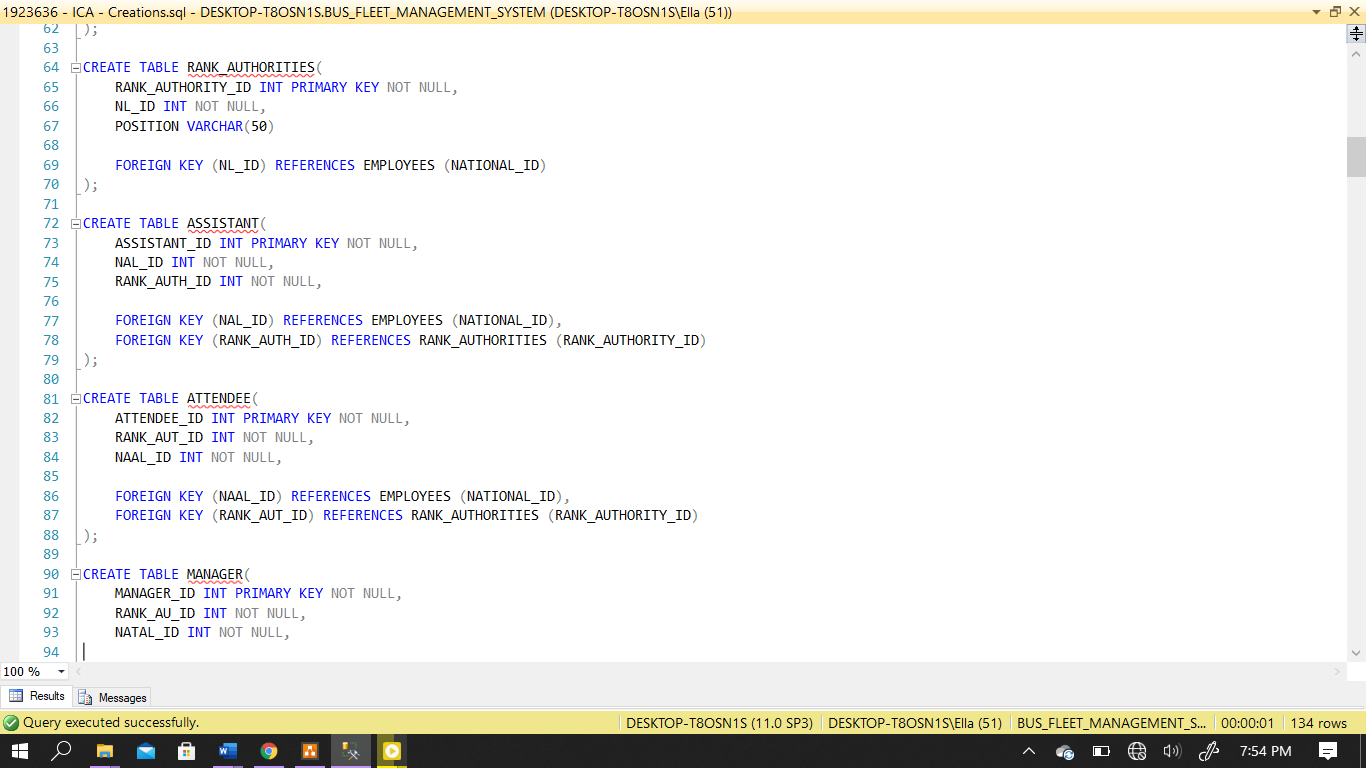
Lines of code 3 to 4 **Creates a database ‘**Bus Fleet management system**’**

Lines 6 to 7 **Uses the database**

The next lines from 13 **Creates tables** in the database

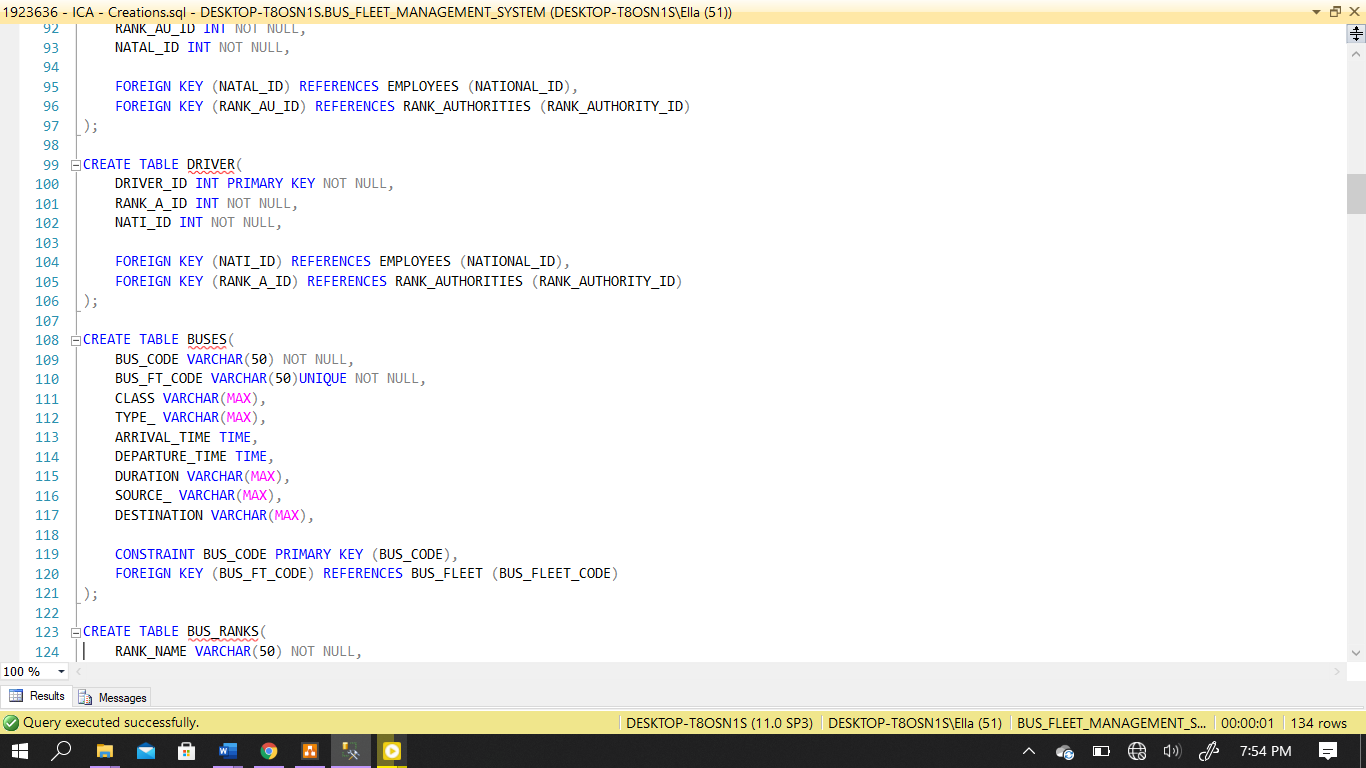
****

After declaring the columns and the data types, a foreign key is created as well. Some columns are directly created as the primary key

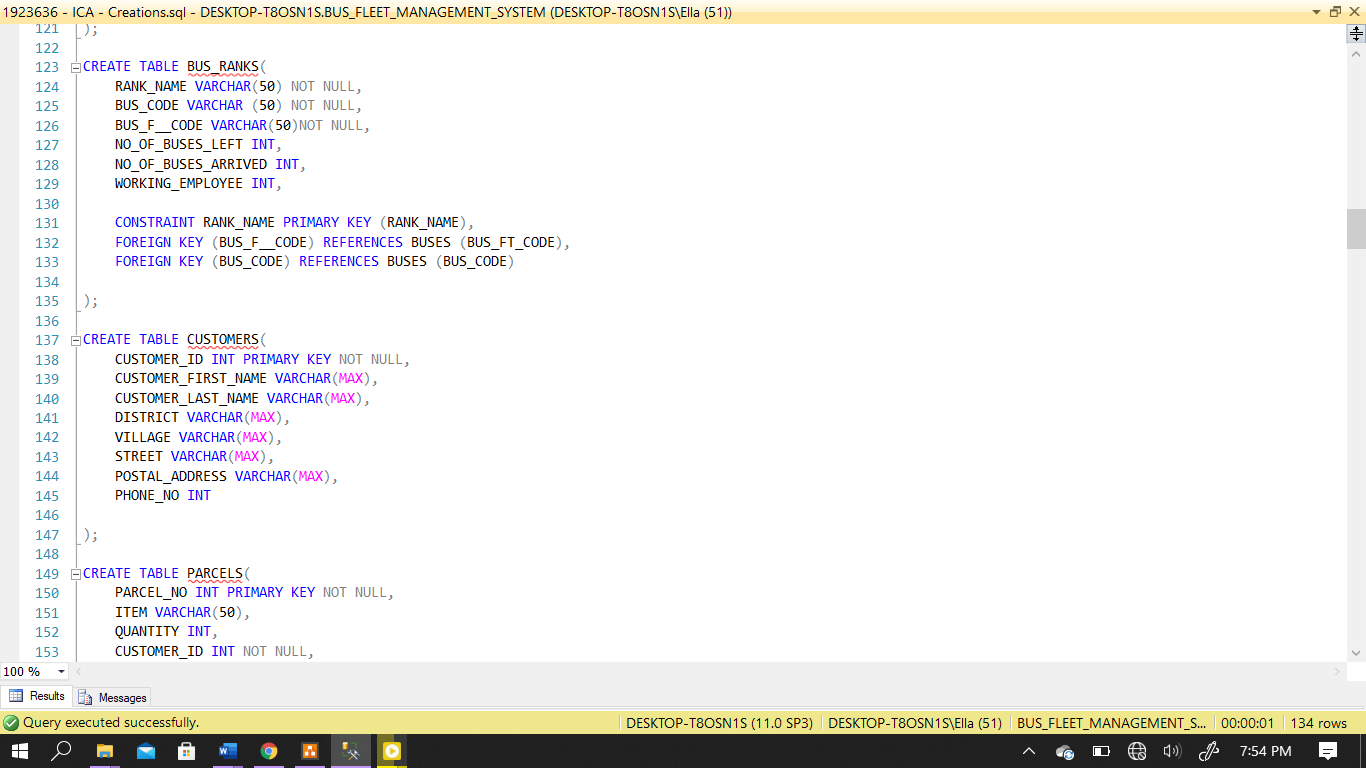
****

Lines of code: 64 to 124

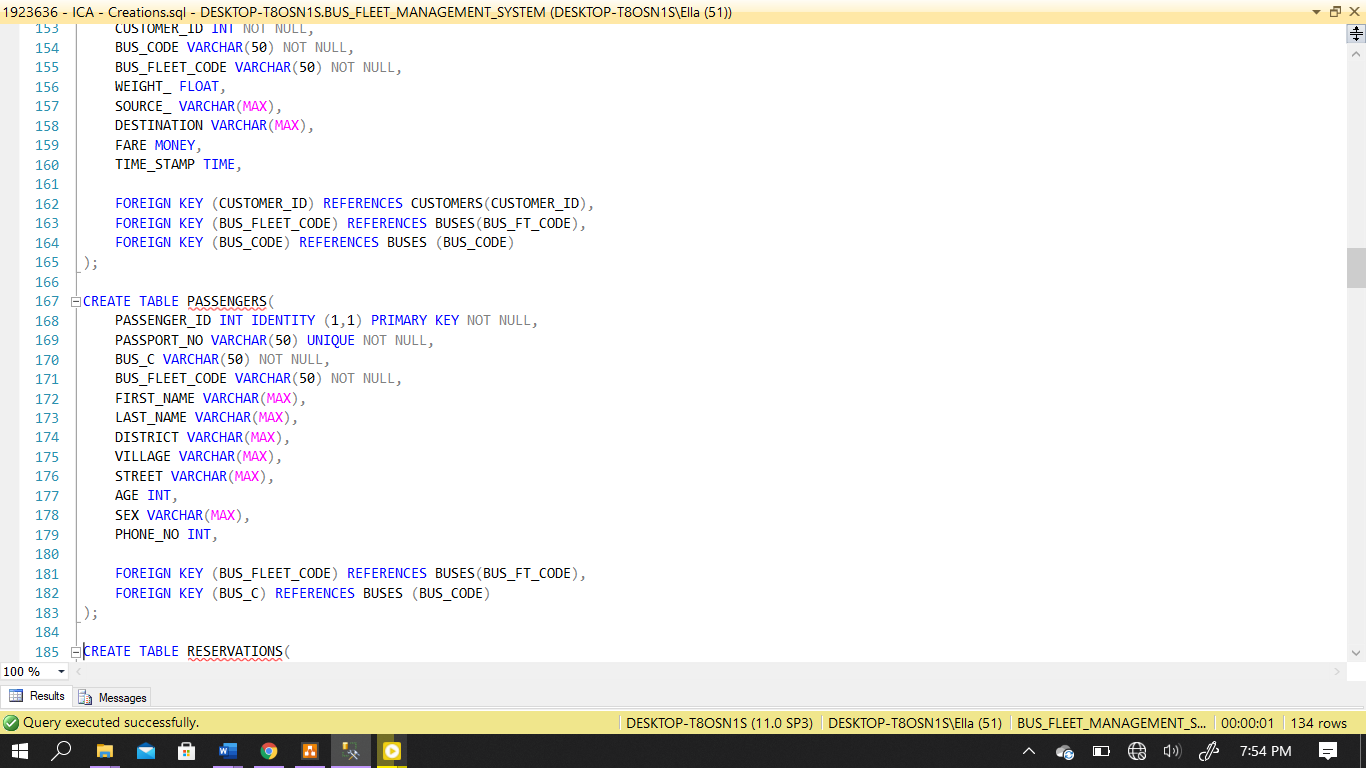
Assistant has two foreign keys

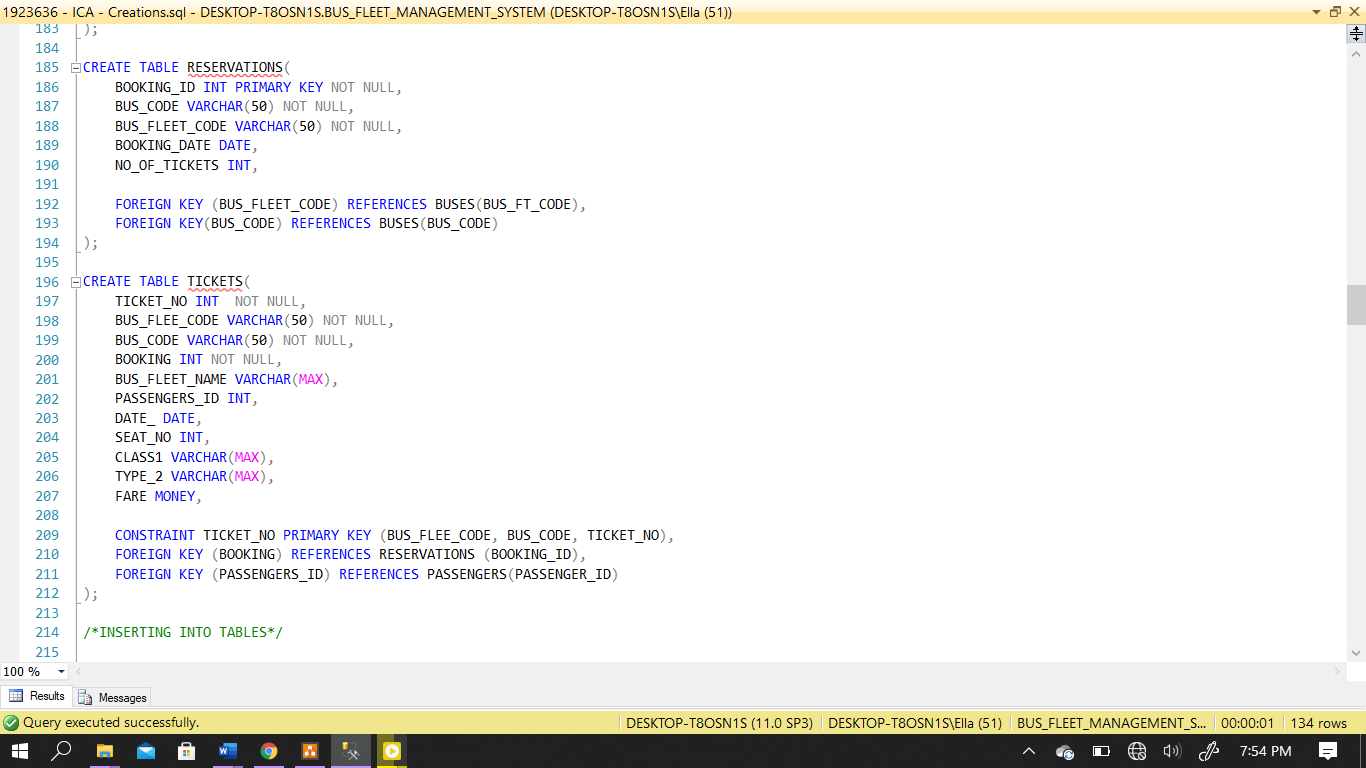
****

Here unique is for the candidate key

****

Lines of code: 122 to 185

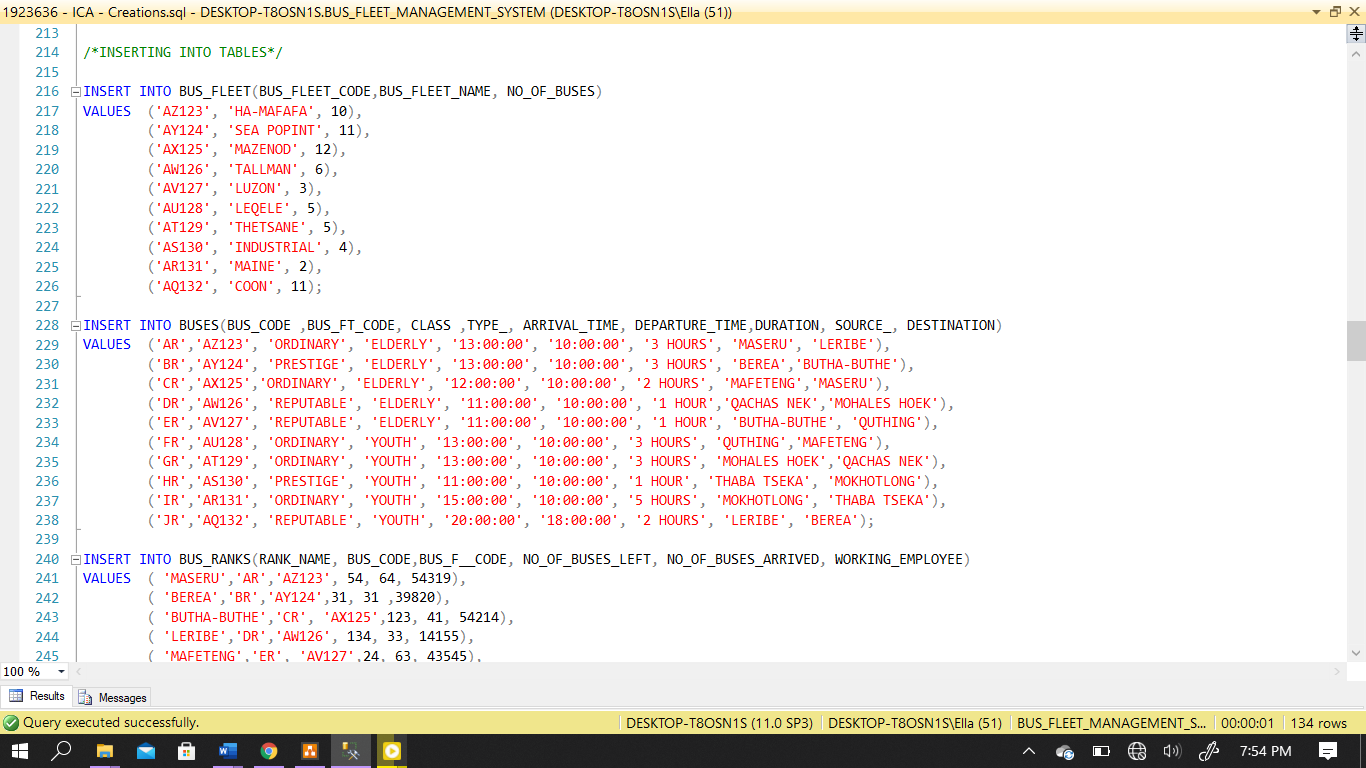
****

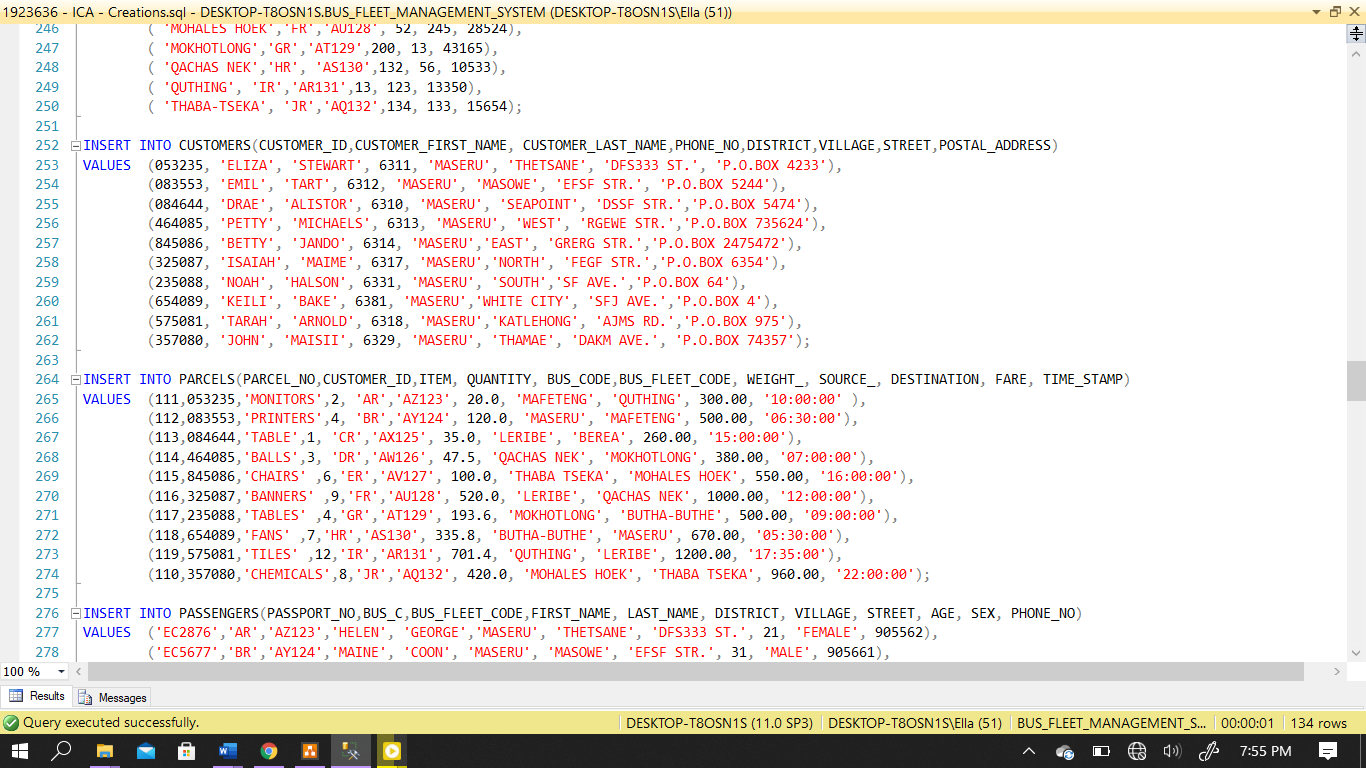
****

Lines of code: 185 to 245

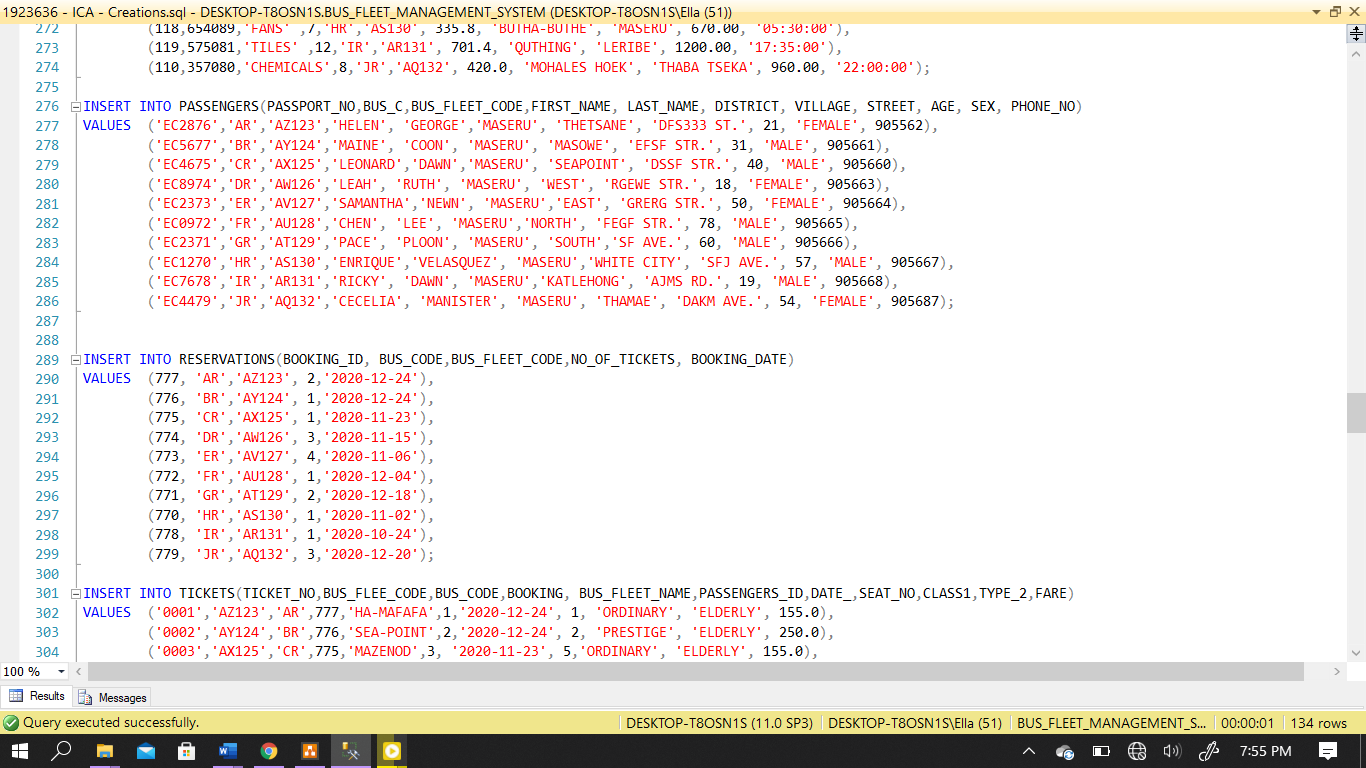
The ticket number has a primary key consisting of three attributes

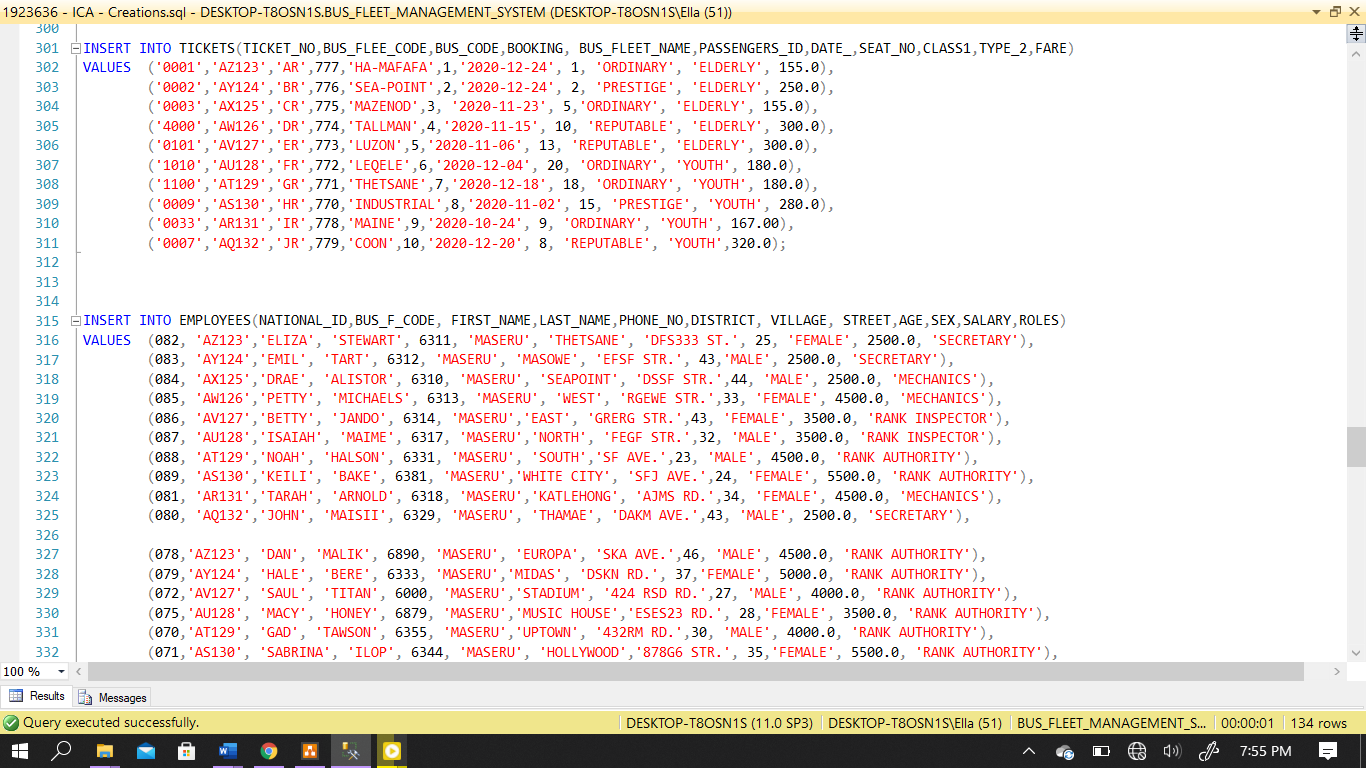
Start of insertion

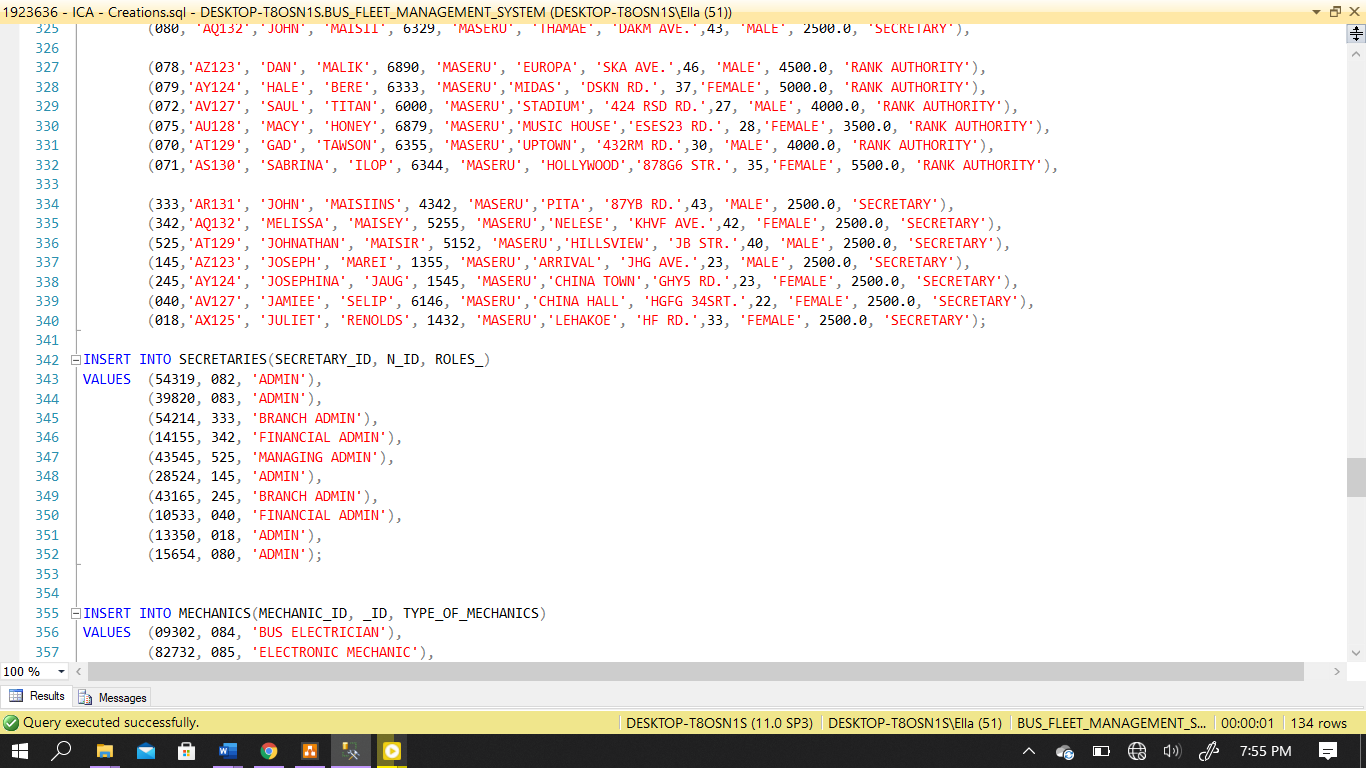
****

****

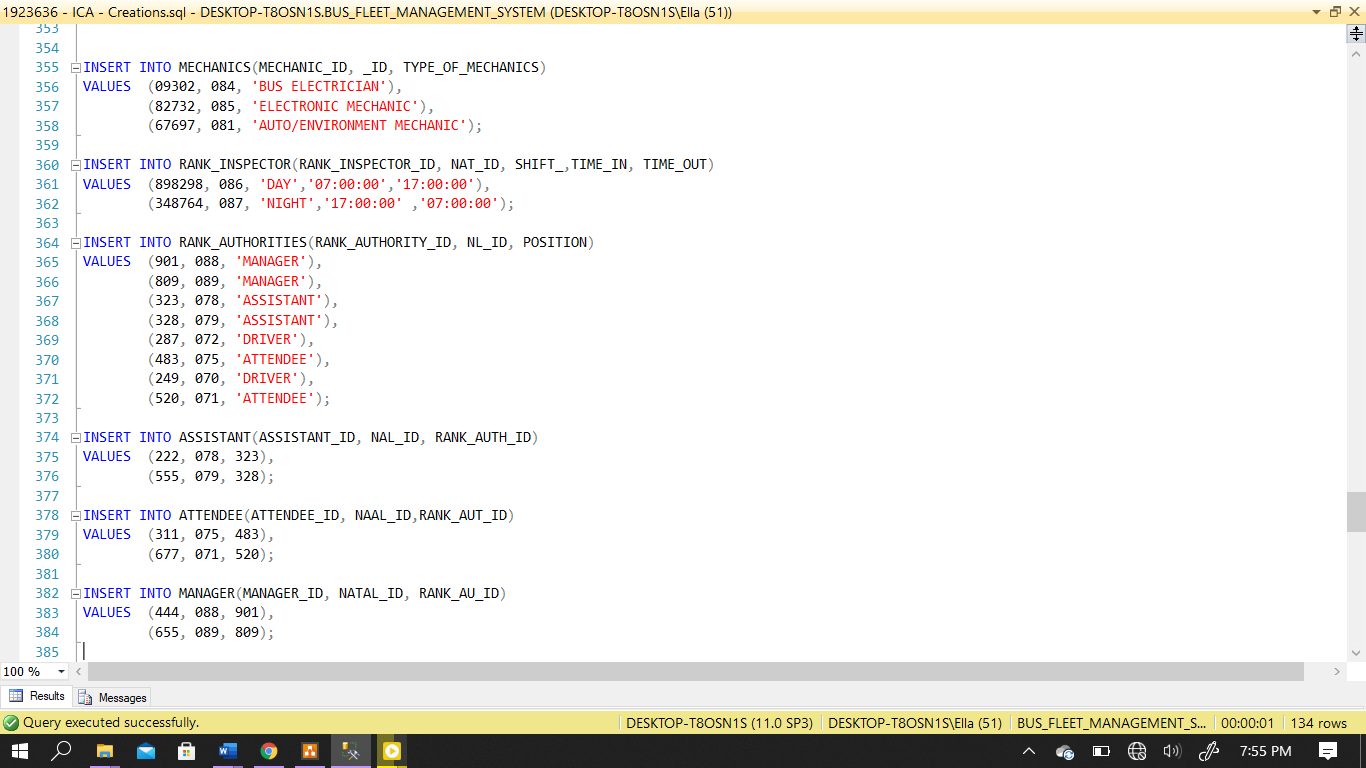
Lines of code: 246 to 304

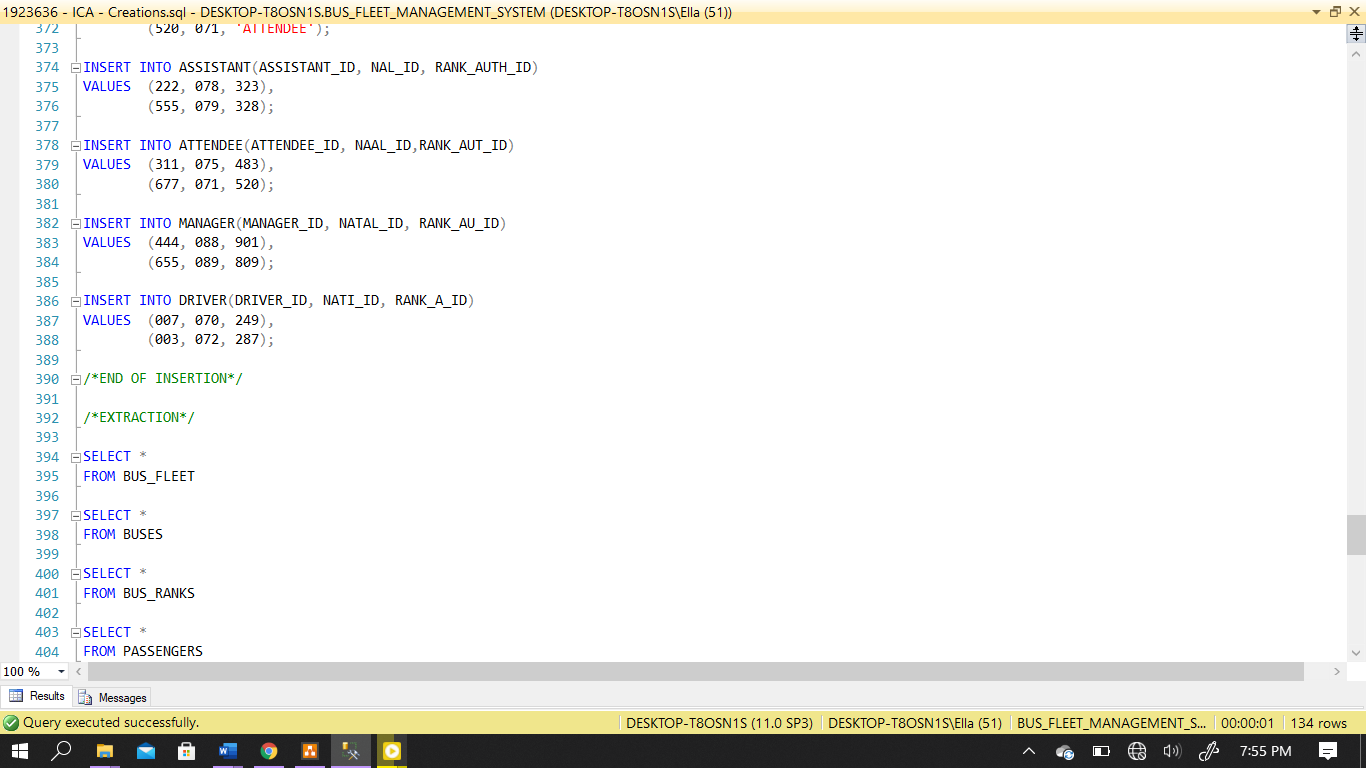
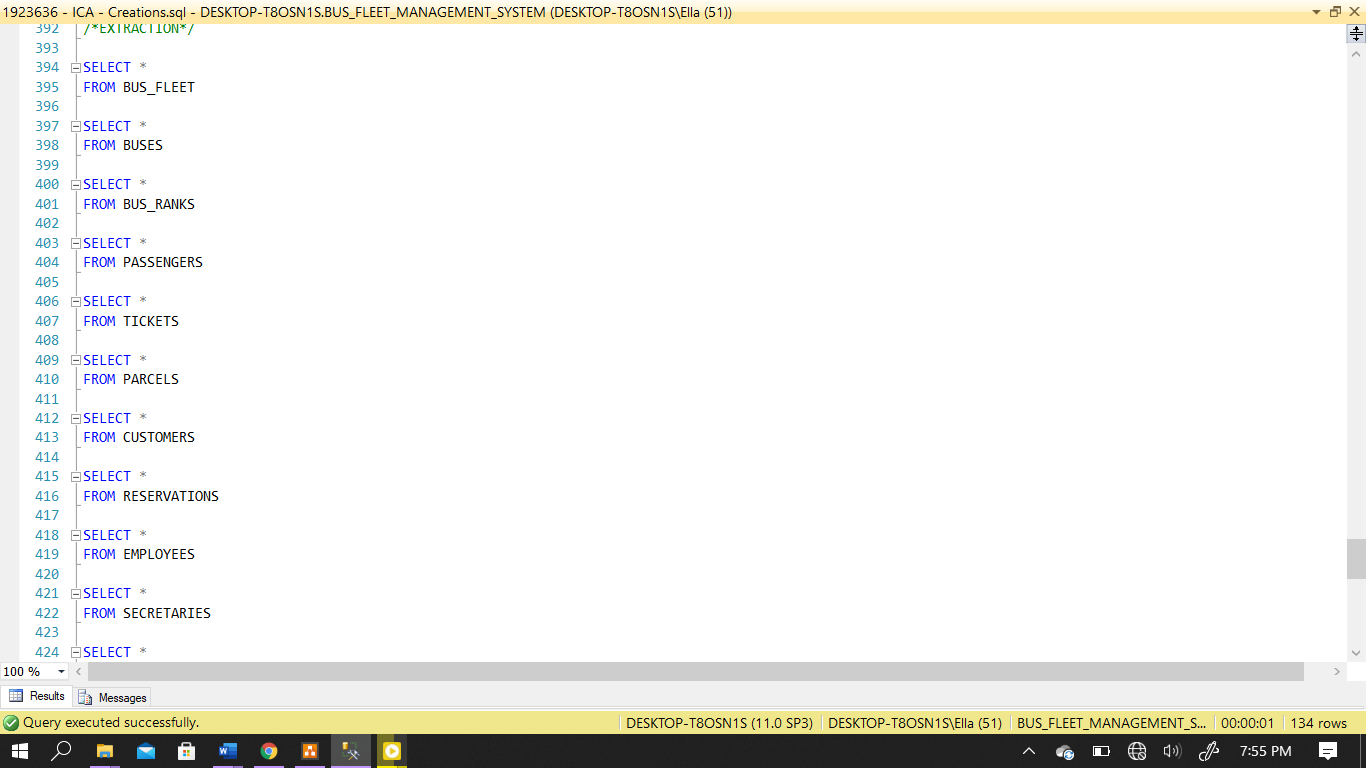
****

****

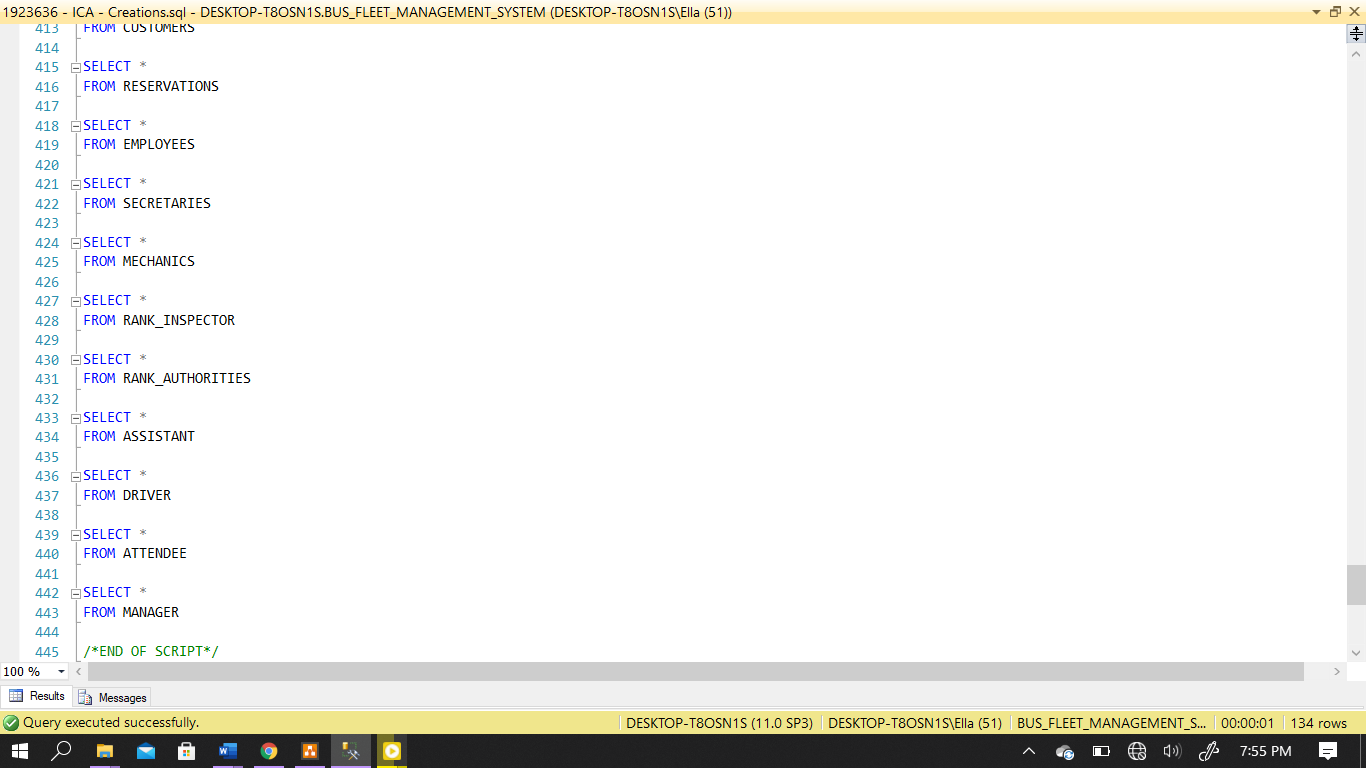
****

Lines of code 304 to 377

****

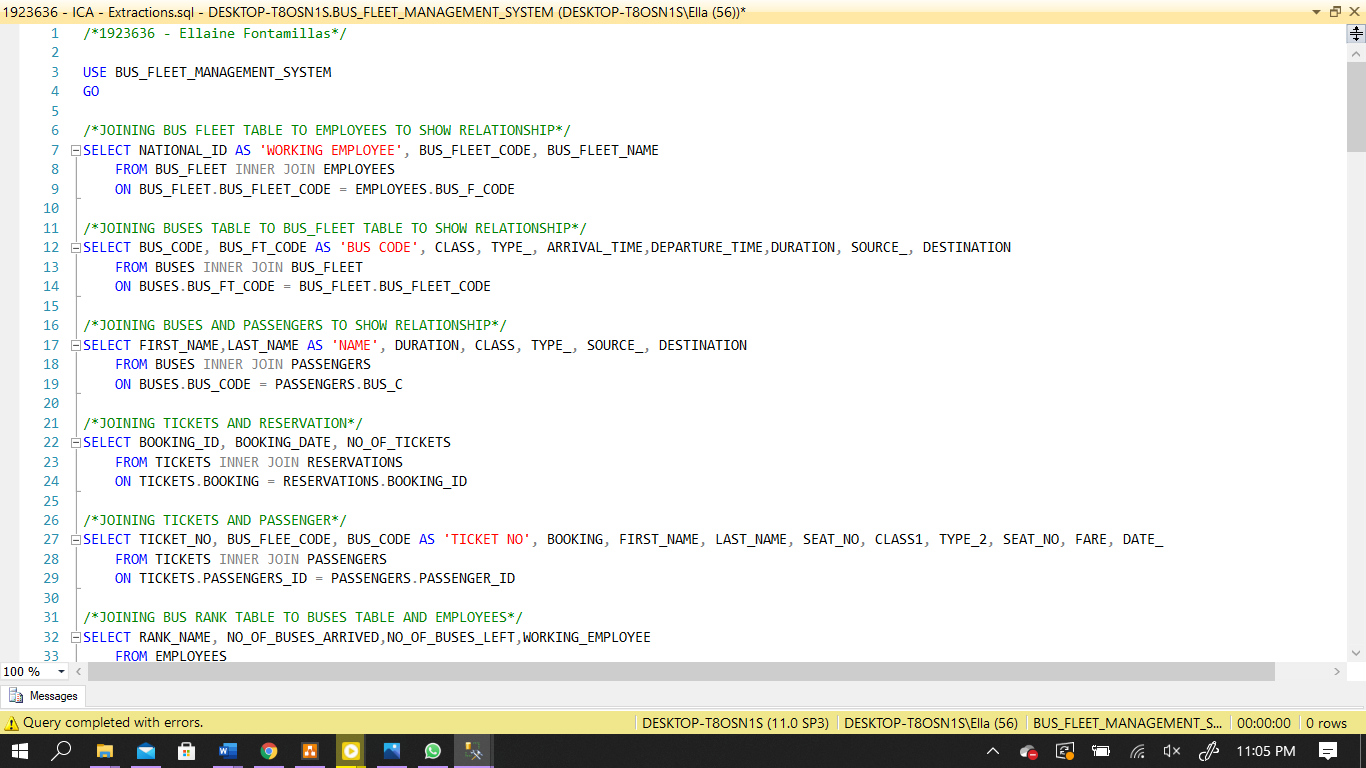
****

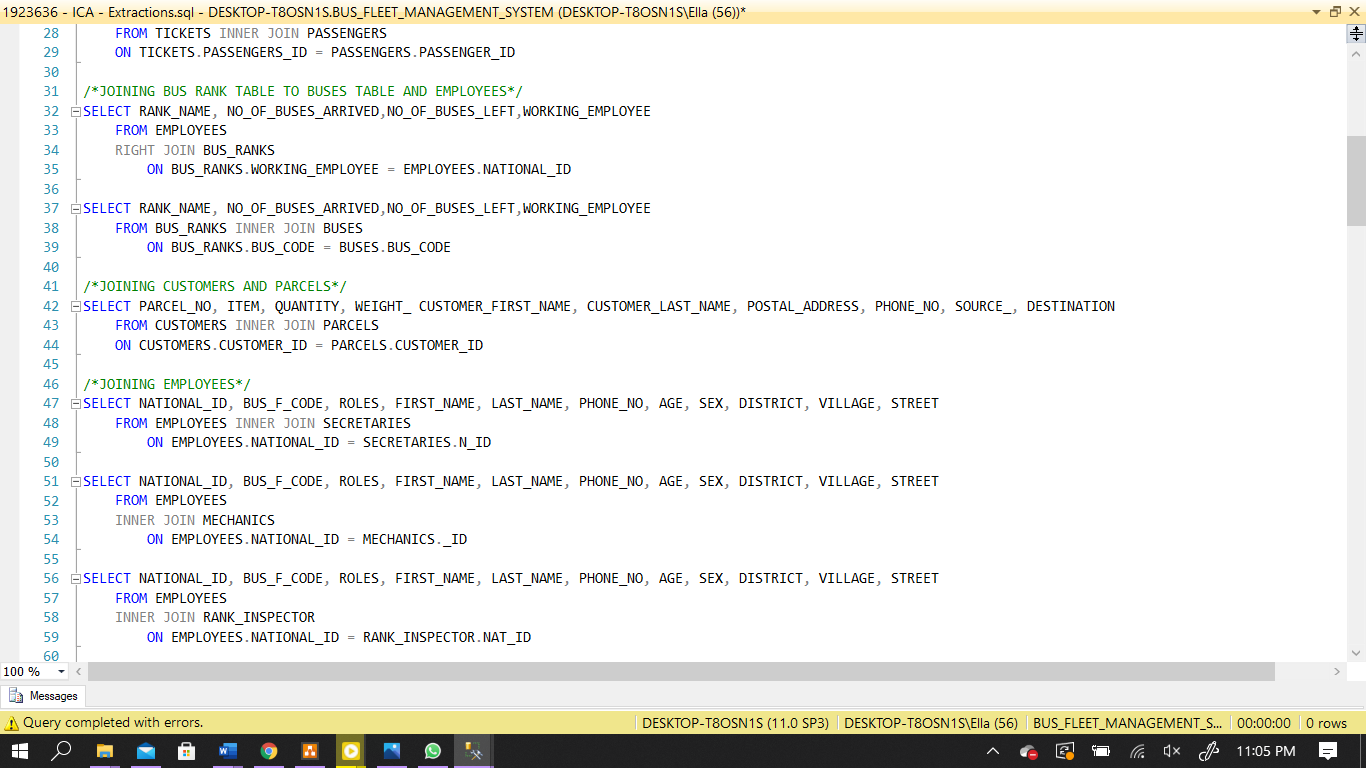
Lines of code 378 to 445

****

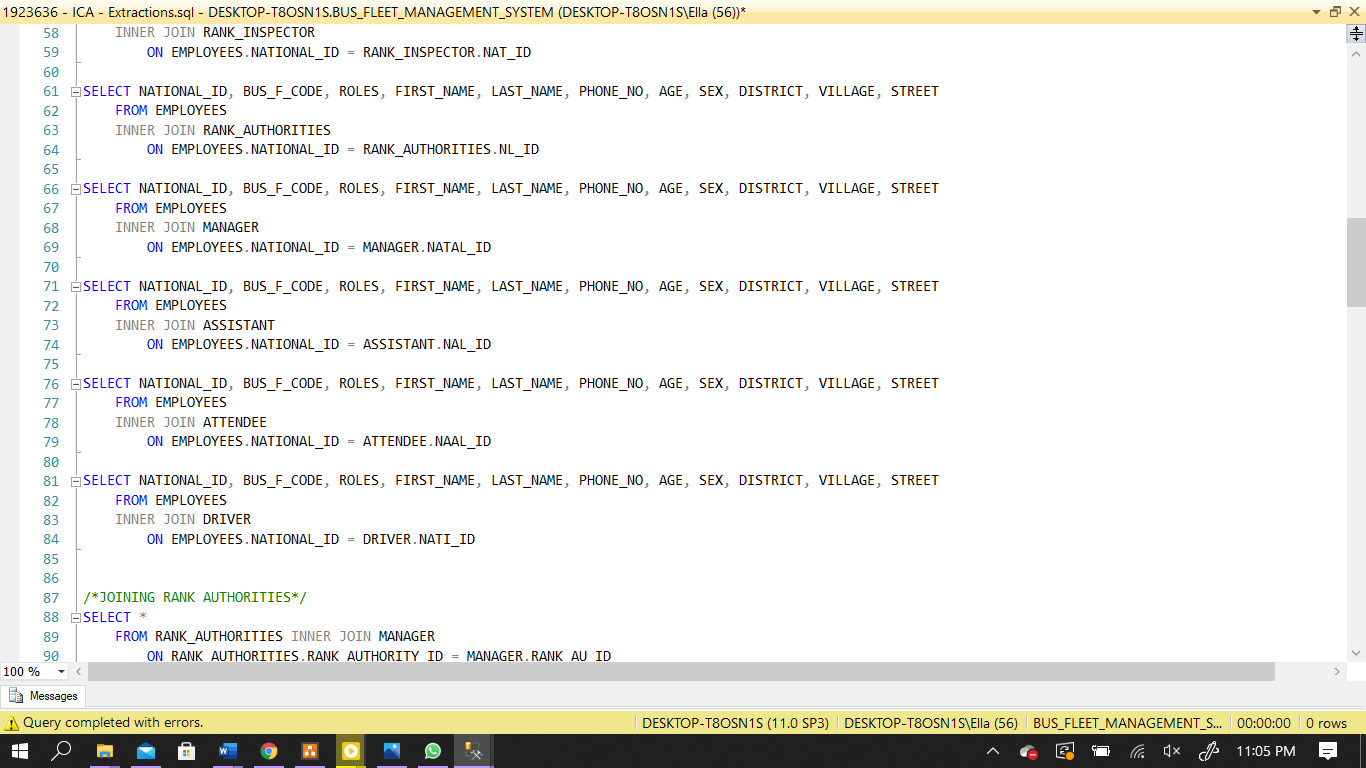
Simple select statements to view the tables

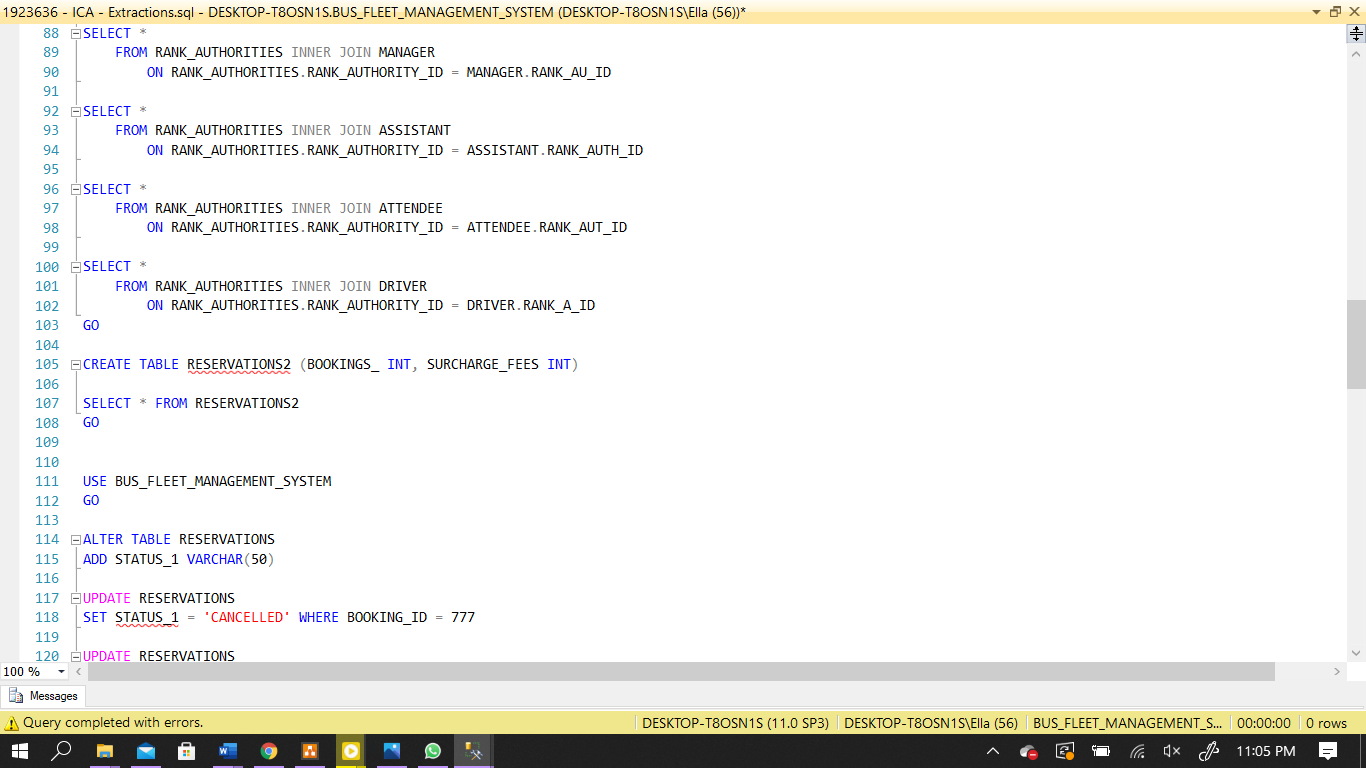
(Normalizing with Entity Relationship Diagramming, n.d.)

1. **Stored Procedures**

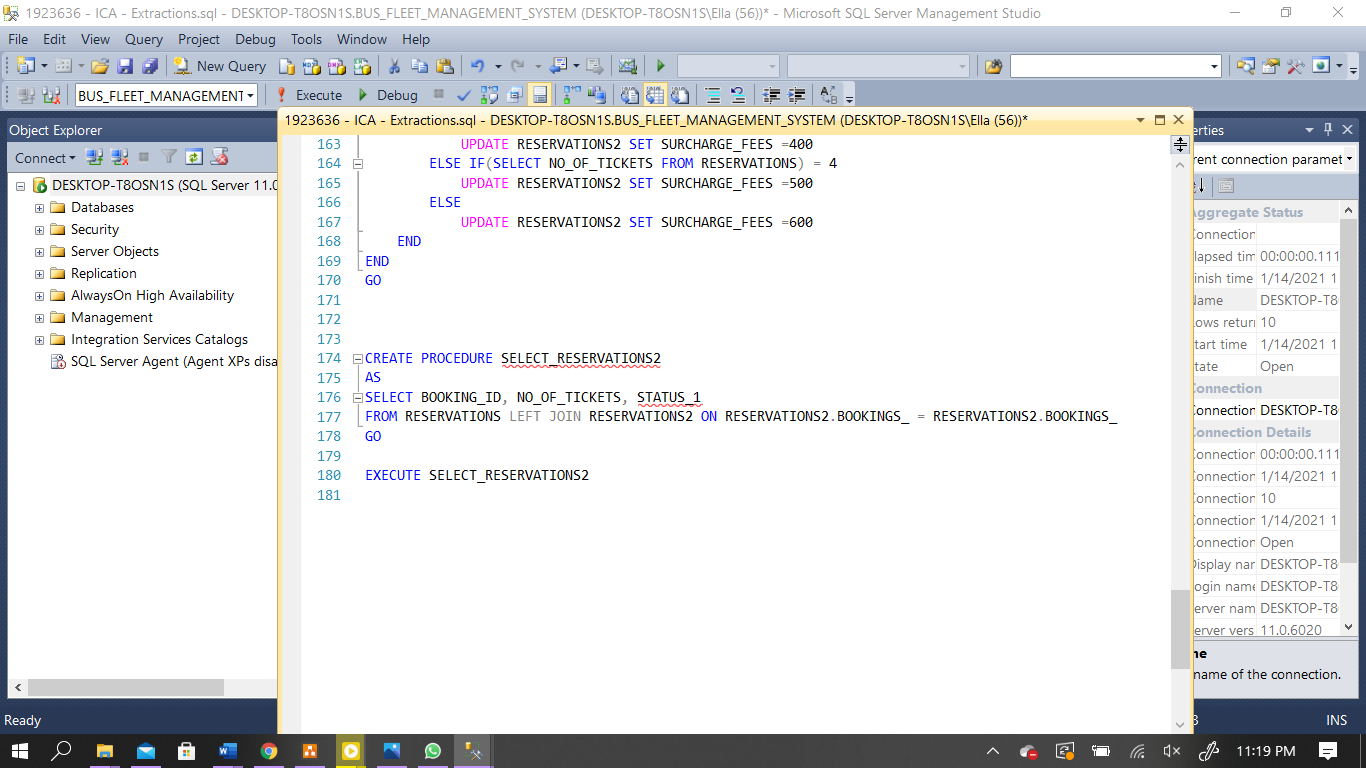
****

I have used the appropriate **JOIN** to join the tables together

****

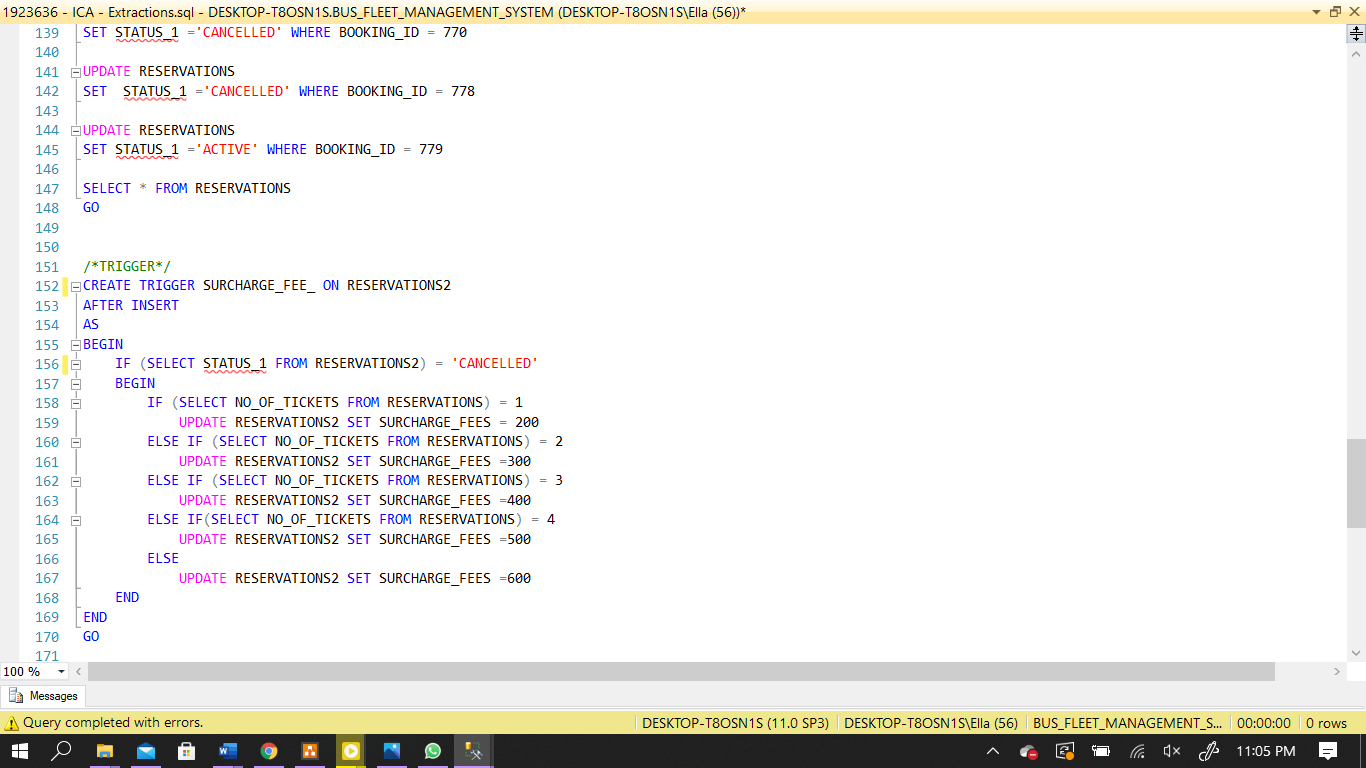
****

This is a table to help with the stored procedure below – it belongs at line code 174

****

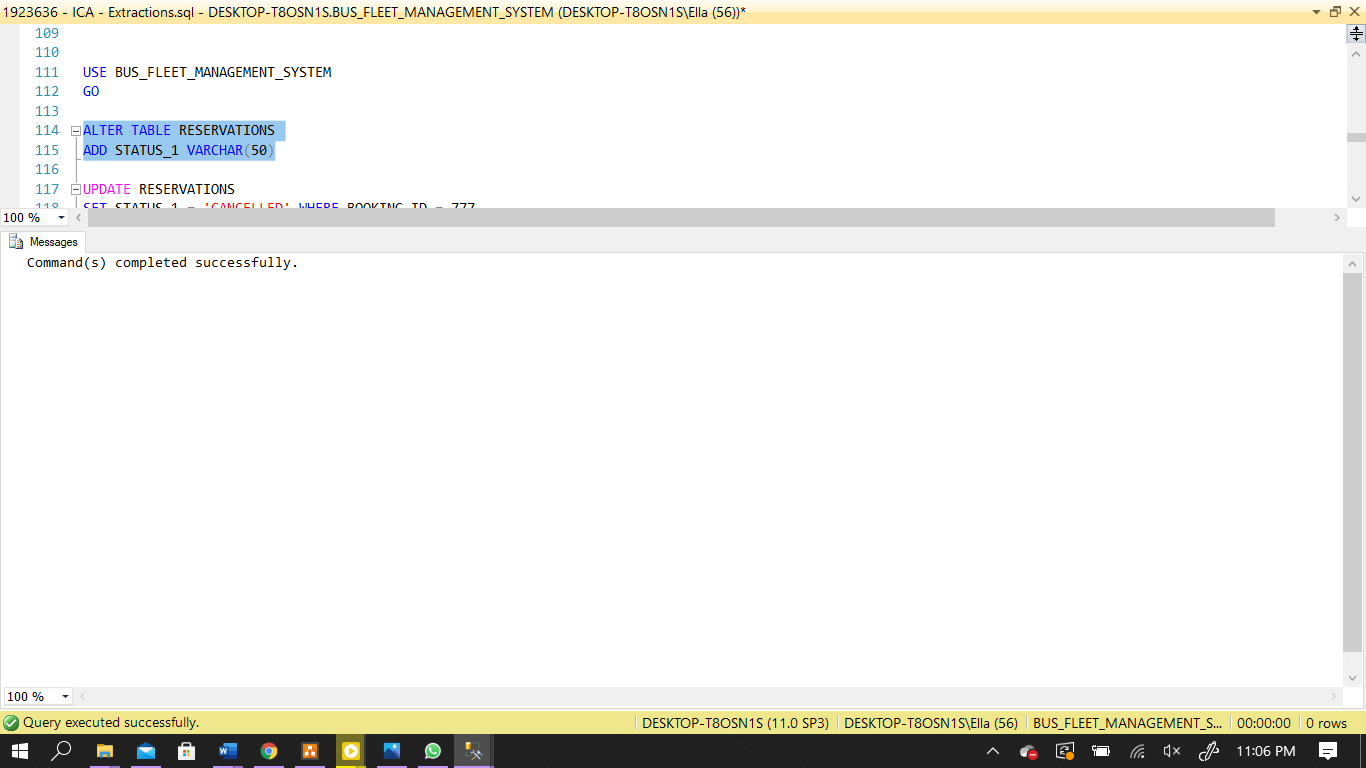
****

1. **Triggers**

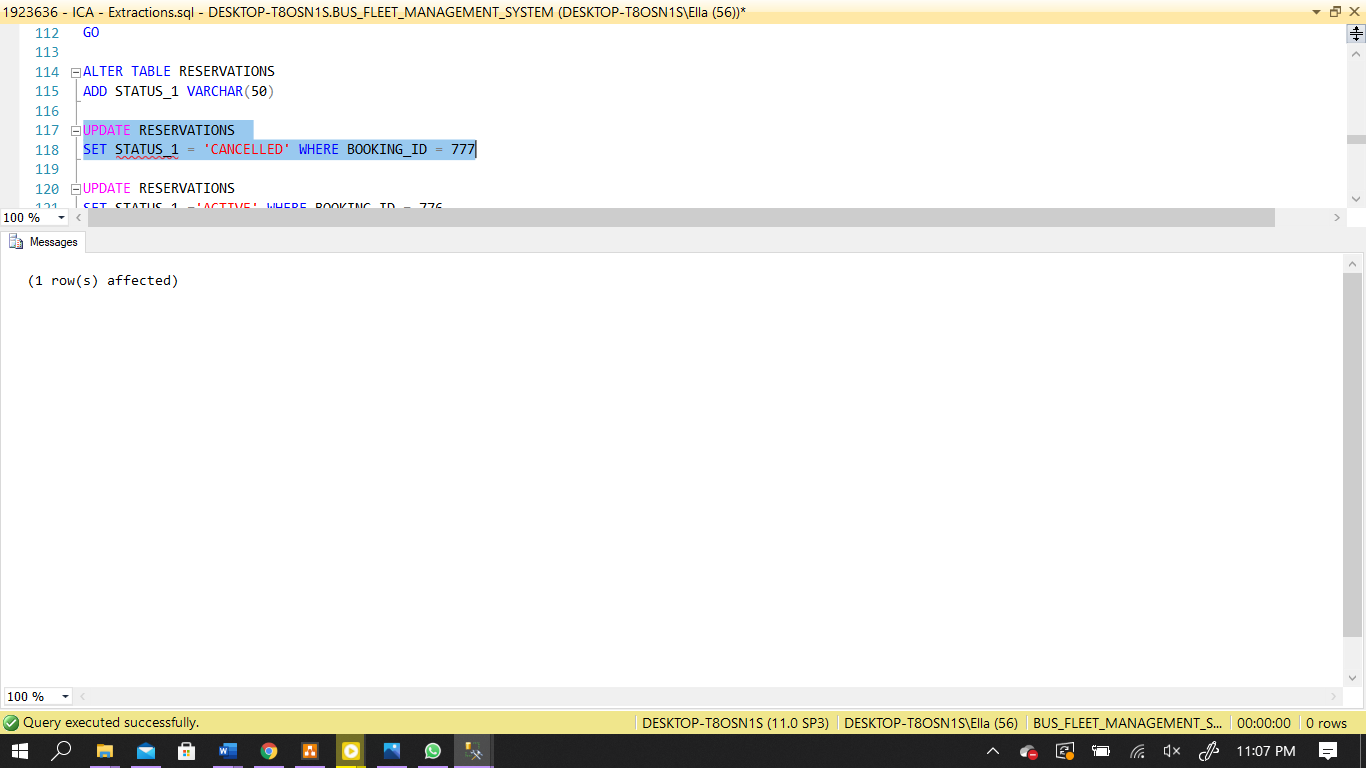
****

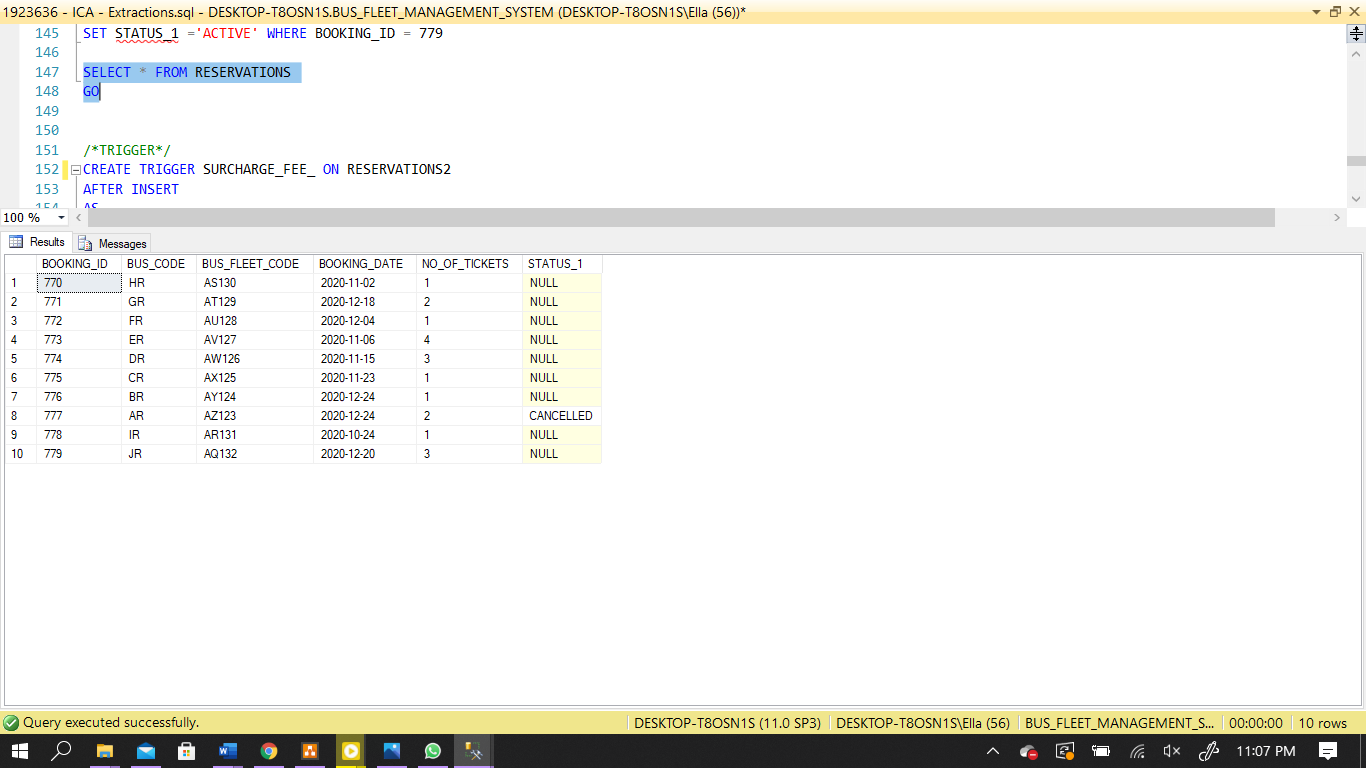
This trigger is for inserting into reservations tables the surcharge fee after updatiion when it checks to see if status1 is of the value cancelled.

It does this after inserting from the code above it

****

****

****

****

1. **Business Rules**

Identification

It is necessary for entities to have an ID for the sake of management and safety of the environment, surrounding people, and the business. It will be required in security checks

Passengers

* Passengers must first purchase a ticket to be granted access to board the bus
* They are allowed to purchase multiple tickets and can cancel the ticket. However, they will need to pay a surcharge fee

Tickets

* Seats of a bus are reserved by ordering tickets
* Depending on the passenger, the tickets are assigned for different buses (type and class)
* Tickets may be cancelled
* Cancelled tickets have cancellation date and surcharge fee

Employees

* Employees are hired to run the bus fleet
* They can be of any hierarchy between mechanic, secretary, rank inspector, rank authority
* A rank authority can also be with either a driver, assistant, manager or attendee
* Secretaries are there for passenger assistance and other roles like administrative duties assigned to them.
* Rank inspectors are the for maintenance and are scheduled shifts to work.

1. **References**

# References

(n.d.). Retrieved January 10, 2021, from https://tdan.com/normalizing-with-entity-relationship-diagramming/4583

(n.d.). Retrieved January 13, 2021, from https://documentation.progress.com/output/ua/OpenEdge\_latest/index.html#page/dmsdv/candidate-keys.html

(n.d.). Retrieved January 11, 2021, from W3Schools: https://www.w3schools.com/sql/sql\_primarykey.ASP

(n.d.). Retrieved January 14, 2021, from https://www.guru99.com/triggers-pl-sql.html

(n.d.). Retrieved January 13, 2021, from https://www.sqlservertutorial.net/sql-server-basics/sql-server-identity/

(OpenEdge 11.7 Documentation, n.d.)

(SQL PRIMARY KEY Constraint, n.d.)

(Oracle PL/SQL Trigger Tutorial: Instead of Compound [Example], n.d.)

(SQL Server Identity Column By Practical Examples, n.d.)