Purdue Mobile Food Solutions

Database Management System

FINAL REPORT

Group 3

Andrew Davis
Cameron Raywood
Gianni Frasca
Wilson Huang
Xin Li Leong

Table of Contents

I. **Background …………………………………………………………….…. 3**

II. **Introduction ……………………………………………………………. 5**

III. **Data Modeling : Business Scenario & ERD …………………………….. 6**

IV. **Relational Data Model ………………………………………………… 9**

a. Relational Schema………………………………………………… 9

b. Normalization……………………..……………………………… 9

c. Sample Data Description…………………………………..…..10

V. **Commentary to Data Model and Design Choices………………………11**

VI. **SQL Create Table Script ……………………………………………...13**

a. SQL Script…………………………………………….……..……13

b. Table in SQL………………………………………………..…..…15

VII. **SQL Queries ……………………………………………………………18**

I. Background

Purdue Mobile Food Solutions (PMFS) is a big data management firm that is located in West Lafayette, Indiana. Founded as a joint venture among Purdue University students, it provides its customers with advanced intel on many facets of the food truck industry using sophisticated data capture technology. Some of the areas within the mobile food industry in which their services encompass include providing detailed information on the different suppliers, locations, and events that customers can utilize or participate in to expand their business presence. On top of presenting comprehensive data to its clients, PMFS also places a great emphasis on outstanding customer service and works extensively with its clients to ensure they can leverage the data to make improvements to their respective companies or meet desired business objectives. With PMFS continually expanding and engaging with a growing list of clients, it has become apparent to them that improvements must be made to their existing databases to more efficiently consolidate customer and industry data to provide the quality level of service that their name has come to be associated with.

To efficiently consolidate existing data and input future data seamlessly, PMFS is looking to restructure and streamline said data in an enhanced database that can more readily meet their business needs. With this enhanced database, PMFS is also placing considerable importance on security and is taking the necessary steps to ensure that there are no vulnerabilities present that could lead to information being illegitimately obtained. This is particularly imperative for PMFS as all client work is done under confidentiality agreements and is one of the attributes that the company prides itself on. To accomplish both efficiency and security within their new database, efforts will be made to simplify the data, which will allow for errors to be more easily avoided and regular security audits to be more quickly carried out. Through the implementation of a consolidated, simplified, and secured database, PMFS will be able to experience the benefits that well-structured data can bring to more proficiently fulfill day-to-day business operations. Working with PMFS on this project is done in anticipation of the company expanding to different sectors of the food industry in the future, such as restaurants and catering services. Because of this, the different methods used to implement the new database will hopefully lead to a more smooth transition for them based on the knowledge they gained from this project.

II. Introduction

With Purdue Mobile Food Solutions multiple expansions and new clients, it became clear that their current database needed to be improved upon to keep up with the increase in customer and industry data. Our main objective for this project is to consolidate existing data into an enhanced database that will provide flawless data entry and retrieval for any amount of data. To meet PMFS business needs we focused on a few objectives being, accuracy and accessibility, security, and scalability.

We decided that accuracy and accessibility were necessary objectives for the database so that PMFS can provide reliable data to its clients free of redundancy and errors but also make it easy to use and access data for their business needs. To accomplish a more accessible database we will make efforts to simplify the data and construct the tables to be easy to understand and navigate.

Along with this PMFS stressed that security is very important, so ensuring that all the data is transferred and stored securely is necessary so it cannot be illegitimately obtained. As stated, all the client work is performed under a confidentiality agreement and security is something the company has prided itself in.

Lastly, we want to ensure that the database is scalable. The main reason for our assignment on this task is because of the increase in clients and the potential of broadening sectors. Scalability is of the most importance because PMFS plans on broadening their expertise in the food industry to also include restaurants and catering services. We plan to make this database fulfill their business needs now and for any future business additions and clients they choose to take on.

III. Data Modeling : Business Scenario & ERD

1. Business Scenario

Purdue Mobile Food Solutions (PMFS) is a big data management firm headquartered in West Lafayette, Indiana. The company provides advanced insight to food truck vendors that enables them to make well-informed and analytically-driven business decisions to enhance their position in their respective locations or sectors. Sophisticated data capture on food truck locations, suppliers, events, franchise permits and food trucks themselves allows PMFS to provide clients with an elevated and meaningful view of the entire industry.

Entity & Attribute

- For event, the database stores EventID (primary key), EventName, EventType, Date, Day, Location.

- For foodtruck, the database stores truckID (primary key), TruckName, Sales, Rating, TruckType

- For foodtruck_event, the database stores EventID (primary key), truckID (primary key), EventSales

- For location, the database stores locationID (primary key), zone, areaSize, authorization
  - There are different types of locations: permit and franchise. Each location must fall into one and only one of these location types.
  - For permit, the database stores locationID (primary key), nooflots
  - For franchise, the database stores locationID (primary key), day, starttime, endtime

- For location_foodtruck, the database stores locationID (primary key), truckID (primary key), Referral

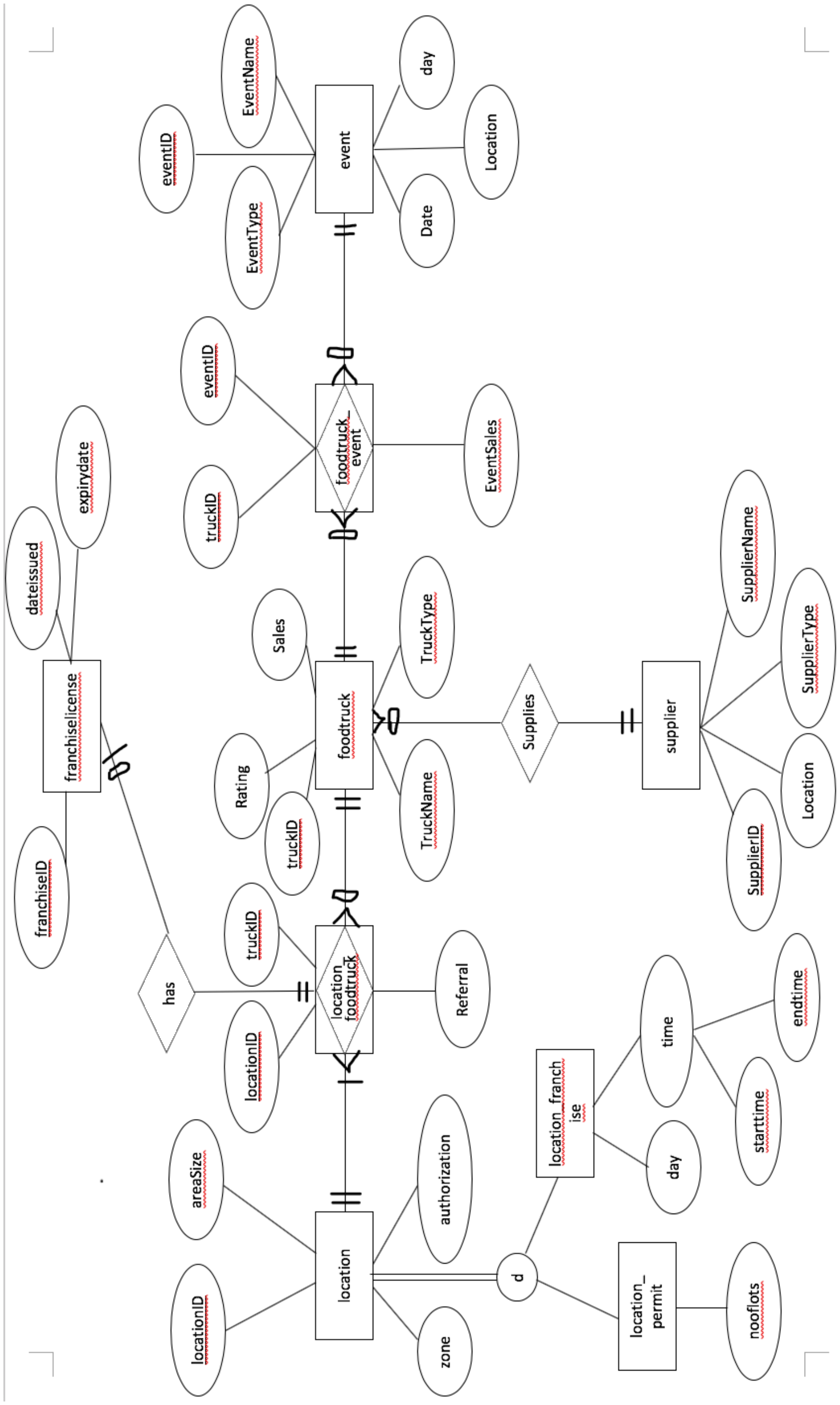
- For supplier, the database stores SupplierID (primary key), Location, SupplierType, SupplierName

- For franchiselicense the database stores franchiseID (primary key), dateissued, expirydate

Relationship

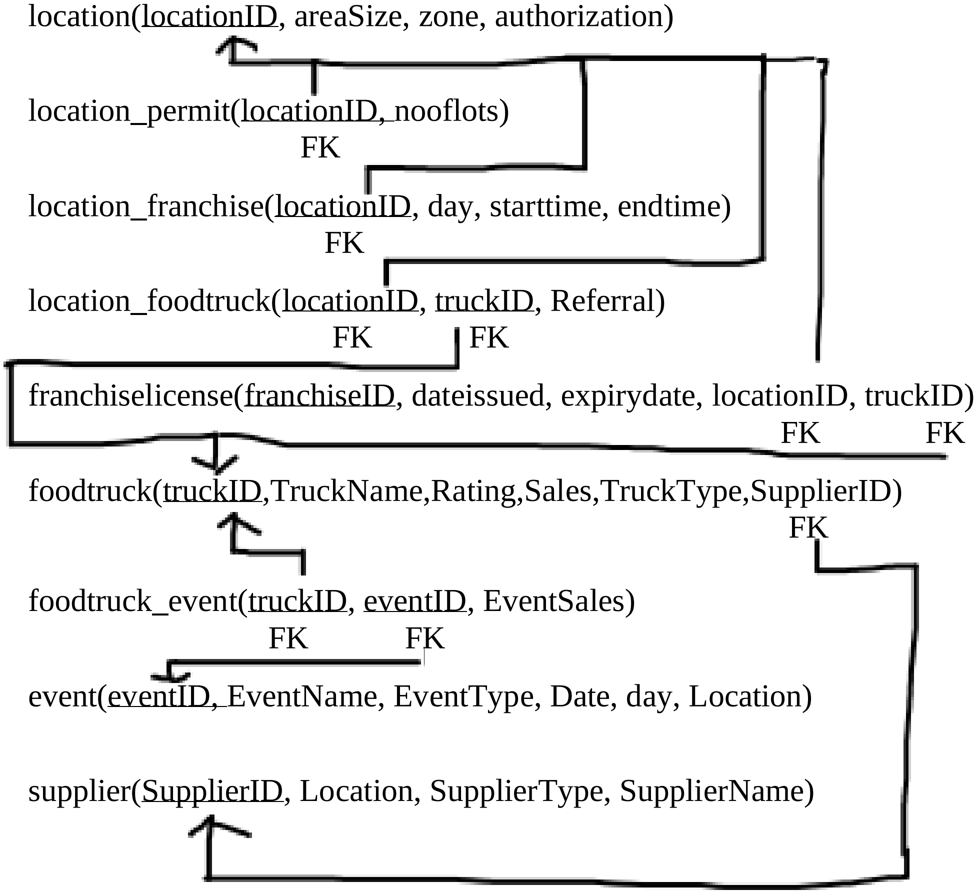
- PMFS has information on a number of potential food truck locations. Each location can be occupied by no food trucks, a single food truck, or many food trucks. A food truck must operate in at least one location, but since it is mobile, it can operate in multiple locations over time. Additionally, every foodtruck that operates in each franchise location must have its unique franchise license, while trucks that are operating in permit locations do not require any special license. Each license is associated with one and only one truck and the specific location its operating in.
- Events are common places in which food trucks are found. At an event, there can be no food trucks or there can be many of them. A particular food truck can choose to operate at no events, or can travel to multiple events.
- Food trucks rely on suppliers to provide them with inventory. Due to contractual agreements, each food truck can have one and only one supplier. A particular supplier can supply no food trucks, a single food truck, or many food trucks.

2. ERD


event
eventID
EventName
EventType
day
Location
Date
foodtruck_event
eventID
truckID
EventSales
foodtruck
Sales
TruckType
Rating
truckID
TruckName
Supplies
supplier
SupplierName
SupplierType
Location
SupplierID
franchiselicense
dateissued
expirydate
franchiseID
has
location_foodtruck
truckID
locationID
Referral
location
areaSize
locationID
authorization
zone
d
location_franchise
time
day
starttime
endtime
location_permit
nooflots

IV. Relational Data Model

1. Relational Schema



location(locationID, areaSize, zone, authorization)

location_permit(locationID, nooflots)
FK

location_franchise(locationID, day, starttime, endtime)
FK

location_foodtruck(locationID, truckID, Referral)
FK FK

franchiselicense(franchiseID, dateissued, expirydate, locationID, truckID)
FK FK

foodtruck(truckID,TruckName,Rating,Sales,TruckType,SupplierID)
FK

foodtruck_event(truckID, eventID, EventSales)
FK FK

event(eventID, EventName, EventType, Date, day, Location)

supplier(SupplierID, Location, SupplierType, SupplierName)

2. Normalization

The above relational schemas already have atomic value in each attribute (no multivalued or composite attributes). There is no partial or transitive functional dependency found. They are currently in 3NF. Therefore, we do not need the process of normalization.

Functional Dependency

locationID -> areaSize, zone, authorization	**FULL**
locationID -> nooflots	**FULL**
locationID -> day, starttime, endtime	**FULL**
locationID, truckID -> Referral	**FULL**

franchiseID -> dateissued, expirydate, locationID, truckID	**FULL**
truckID -> TruckName,Rating,Sales,TruckType,SupplierID	**FULL**
truckID, eventID -> EventSales	**FULL**
eventID -> EventName, EventType, Date, day, Location	**FULL**
SupplierID -> Location, SupplierType, SupplierName	**FULL**

3. Sample Data Description

a) **location**: This table will have 4 attributes: locationID (primary key), zone, areaSize, and authorization.

b) **location_permit**: This table will have 2 attributes: locationID (primary key), nooflots

c) **location_franchise**: This table will have 4 attributes: locationID (primary key), day, starttime, endtime

d) **franchiselicense**: This table will have 5 attributes: locationID (primary key), dateissued, expirydate, locationID (foreign key), truckID (foreign key).

e) **foodtruck**: This table will have 6 attributes: truckID (primary key), Rating, TruckName, TruckType, Sales, and SupplierID(foreign key).

f) **supplier**: This table will have 4 attributes: SupplierID (primary key), SupplierName, SupplierType, and Location.

g) **event**: This table will have 6 attributes: eventID (primary key), EventName, EventType, Date, day, and Location.

h) **location_foodtruck**: This table will have r attributes: locationID (primary key) and truckID (primary key), Referral.

i) **foodtruck_event**: This table will have r attributes: truckID(primary key) and EventID(primary key), EventSales

V. Commentary to Date Model and Design Choices

When we first received Purdue Mobile Food Solutions' database, it had multiple design flaws like data redundancy issues, poor indexes and poor normalization. Thus, we made multiple changes to the database by adding proper indexes in each entity and setting it as the primary key, creating associative entities and also going through the normalization process with their database. This helped to create a more effective and efficient database that can optimize the data analysis process in the future. In our Entity Relationship Diagram, you can see our design choices. Starting from the left side of our ERD, we noticed there was a many-to-many relationship between event and foodtruck, thus we created an associative entity, foodtruck_event, here. Each foodtruck may choose to be at an event or not and can be at multiple events while an event may have no foodtrucks at all or multiple foodtrucks attending. We also decided to record the sales associated with each foodtruck and the event they attended under the associative entity. Additionally, because each truck that operates at a franchise location requires a franchise license we connected franchiselicenses to this associative entity. Since the associate entity has trucks that operates in permit and franchise location, not all trucks and location combination needs a license and can only have one license. Each license is also unique to one and only one combination of truck and the location it's operating in the associative entity. Moving on to the right side of the ERD, we also noted a many-to-many relationship between foodtruck and location and created an associative entity, locaiton_foodtruck. Each foodtruck must at least operate at one location, but can operate at multiple locations while a location may not have any foodtrucks operating or have multiple foodtrucks operating there. In our associative entity we also recorded how each foodtruck was referred to the specific location that they’re operating in. Lastly, as you move toward the bottom of our ERD, you can see an optional one-to-many relationship between foodtruck and supplier. Each foodtruck can have one and only one supplier while each supplier may not be supplying anything to any foodtruck, the supplier may supply ingredients to many foodtrucks

VI. SQL Create Table Script

1. SQL Script

   a. **event**

   ```
   CREATE TABLE event (
     EventID varchar(20) NOT NULL DEFAULT '',
     EventName text,
     EventType text,
     Date datetime DEFAULT NULL,
     Day text,
     Location text,
     PRIMARY KEY (EventID));
   ```

   b. **foodtruck_event**

   ```
   CREATE TABLE foodtruck_event (
     EventID varchar(20) NOT NULL DEFAULT '',
     truckID varchar(20) NOT NULL DEFAULT '',
     EventSales int(11) DEFAULT NULL,
     PRIMARY KEY (truckID,EventID));
   ```

   c. **franchiselicense**

   ```
   CREATE TABLE franchiselicense (
     franchiseID varchar(20) NOT NULL DEFAULT '',
     locationID varchar(20) DEFAULT NULL,
     truckID varchar(20) DEFAULT NULL,
     dateissued datetime DEFAULT NULL,
     expirydate datetime DEFAULT NULL,
     PRIMARY KEY(franchiseID));
   ```

   d. **location**

   ```
   CREATE TABLE location (
     locationID varchar(20) NOT NULL DEFAULT '',
     zone text,
     areaSize int(11) DEFAULT NULL,
     authorization text,
     PRIMARY KEY (locationID));
   ```

   e. **location_foodtruck**

   ```
   CREATE TABLE location_foodtruck (
     locationID varchar(20) NOT NULL DEFAULT '',
     truckID varchar(20) NOT NULL DEFAULT '',
   ```

```
  Referral text,
  PRIMARY KEY (truckID,locationID));
```

f. **location_franchise**

```
CREATE TABLE location_franchise (
  locationID varchar(20) NOT NULL,
  day text,
  starttime datetime DEFAULT NULL,
  endtime datetime DEFAULT NULL,
  PRIMARY KEY (locationID));
```

g. **location_permit**

```
CREATE TABLE location_permit (
  locationID varchar(20) NOT NULL,
  nooflots int(11) DEFAULT NULL,
  PRIMARY KEY (locationID));
```

h. **supplier**

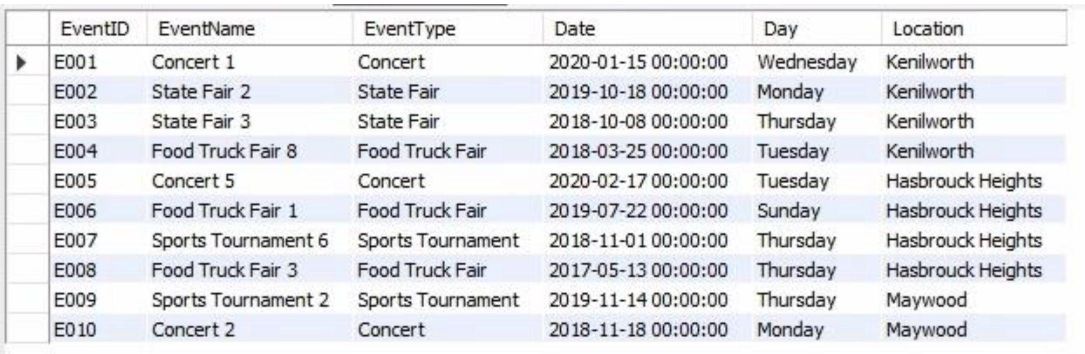
```
CREATE TABLE supplier (
  SupplierID varchar(20) NOT NULL DEFAULT '',
  Location text,
  SupplierType text,
  SupplierName text,
  PRIMARY KEY (SupplierID));
```

i. **foodtruck**

```
CREATE TABLE foodtruck (
  truckID varchar(20) NOT NULL DEFAULT '',
  TruckName text,
  Sales int(11) DEFAULT NULL,
  Rating int(11) DEFAULT NULL,
  TruckType text,
  SupplierID varchar(20) DEFAULT NULL,
  PRIMARY KEY (truckID),
  KEY FK_foodtruck_supplier (SupplierID),
  CONSTRAINT FK_foodtruck_supplier FOREIGN KEY (SupplierID)
REFERENCES supplier (SupplierID));
```

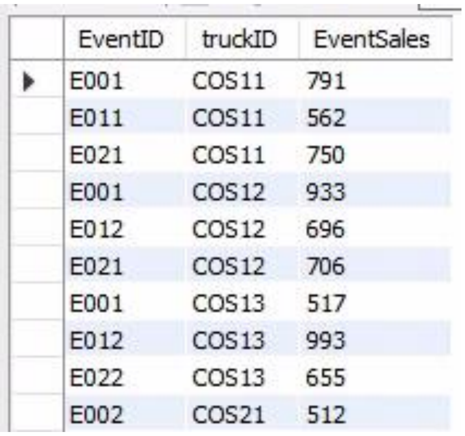
2. Table in SQL

a. **event**



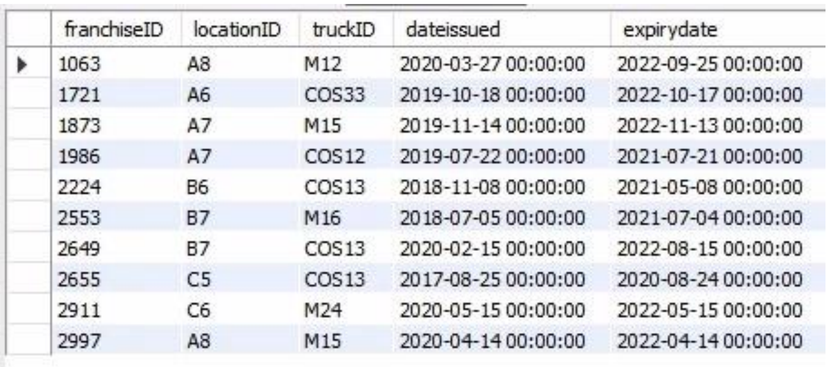
EventID	EventName	EventType	Date	Day	Location
E001	Concert 1	Concert	2020-01-15 00:00:00	Wednesday	Kenilworth
E002	State Fair 2	State Fair	2019-10-18 00:00:00	Monday	Kenilworth
E003	State Fair 3	State Fair	2018-10-08 00:00:00	Thursday	Kenilworth
E004	Food Truck Fair 8	Food Truck Fair	2018-03-25 00:00:00	Tuesday	Kenilworth
E005	Concert 5	Concert	2020-02-17 00:00:00	Tuesday	Hasbrouck Heights
E006	Food Truck Fair 1	Food Truck Fair	2019-07-22 00:00:00	Sunday	Hasbrouck Heights
E007	Sports Tournament 6	Sports Tournament	2018-11-01 00:00:00	Thursday	Hasbrouck Heights
E008	Food Truck Fair 3	Food Truck Fair	2017-05-13 00:00:00	Thursday	Hasbrouck Heights
E009	Sports Tournament 2	Sports Tournament	2019-11-14 00:00:00	Thursday	Maywood
E010	Concert 2	Concert	2018-11-18 00:00:00	Monday	Maywood

b. **foodtruck_event**



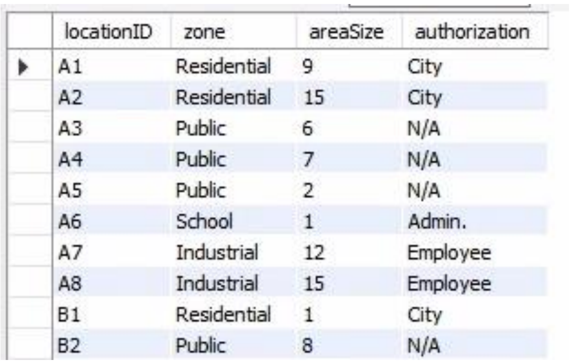
EventID	truckID	EventSales
E001	COS11	791
E011	COS11	562
E021	COS11	750
E001	COS12	933
E012	COS12	696
E021	COS12	706
E001	COS13	517
E012	COS13	993
E022	COS13	655
E002	COS21	512

c. **franchiselicense**



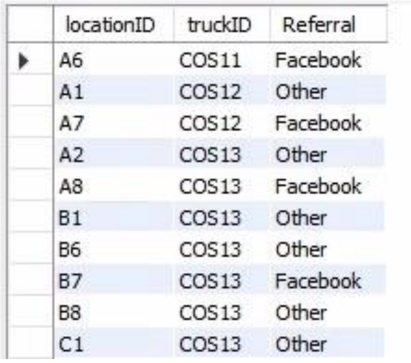
franchiseID	locationID	truckID	dateissued	expirydate
1063	A8	M12	2020-03-27 00:00:00	2022-09-25 00:00:00
1721	A6	COS33	2019-10-18 00:00:00	2022-10-17 00:00:00
1873	A7	M15	2019-11-14 00:00:00	2022-11-13 00:00:00
1986	A7	COS12	2019-07-22 00:00:00	2021-07-21 00:00:00
2224	B6	COS13	2018-11-08 00:00:00	2021-05-08 00:00:00
2553	B7	M16	2018-07-05 00:00:00	2021-07-04 00:00:00
2649	B7	COS13	2020-02-15 00:00:00	2022-08-15 00:00:00
2655	C5	COS13	2017-08-25 00:00:00	2020-08-24 00:00:00
2911	C6	M24	2020-05-15 00:00:00	2022-05-15 00:00:00
2997	A8	M15	2020-04-14 00:00:00	2022-04-14 00:00:00

d. **location**



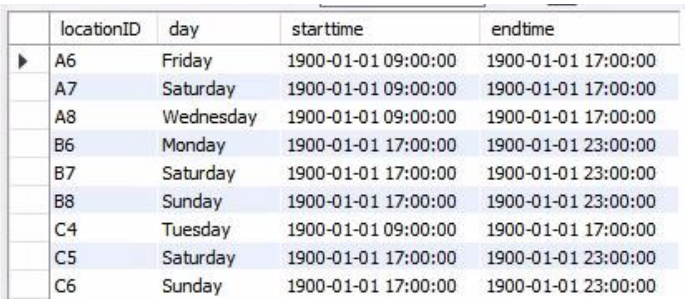
locationID	zone	areaSize	authorization
A1	Residential	9	City
A2	Residential	15	City
A3	Public	6	N/A
A4	Public	7	N/A
A5	Public	2	N/A
A6	School	1	Admin.
A7	Industrial	12	Employee
A8	Industrial	15	Employee
B1	Residential	1	City
B2	Public	8	N/A

e. **location_foodtruck**



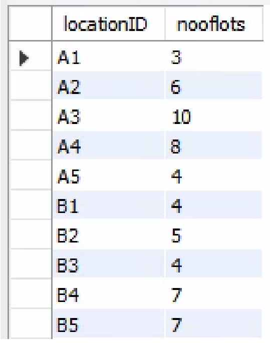
locationID	truckID	Referral
A6	COS11	Facebook
A1	COS12	Other
A7	COS12	Facebook
A2	COS13	Other
A8	COS13	Facebook
B1	COS13	Other
B6	COS13	Other
B7	COS13	Facebook
B8	COS13	Other
C1	COS13	Other

f. **location_franchise**



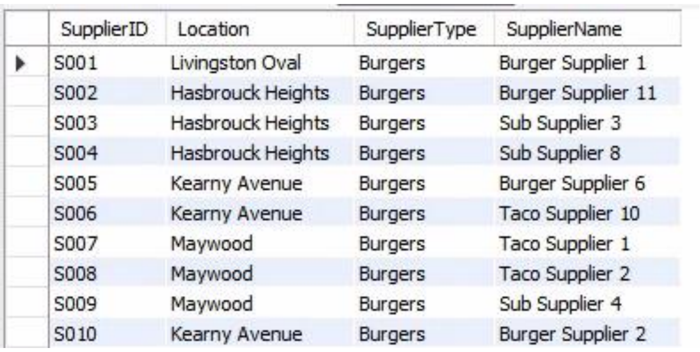
locationID	day	starttime	endtime
A6	Friday	1900-01-01 09:00:00	1900-01-01 17:00:00
A7	Saturday	1900-01-01 09:00:00	1900-01-01 17:00:00
A8	Wednesday	1900-01-01 09:00:00	1900-01-01 17:00:00
B6	Monday	1900-01-01 17:00:00	1900-01-01 23:00:00
B7	Saturday	1900-01-01 17:00:00	1900-01-01 23:00:00
B8	Sunday	1900-01-01 17:00:00	1900-01-01 23:00:00
C4	Tuesday	1900-01-01 09:00:00	1900-01-01 17:00:00
C5	Saturday	1900-01-01 17:00:00	1900-01-01 23:00:00
C6	Sunday	1900-01-01 17:00:00	1900-01-01 23:00:00

g. **location_permit**



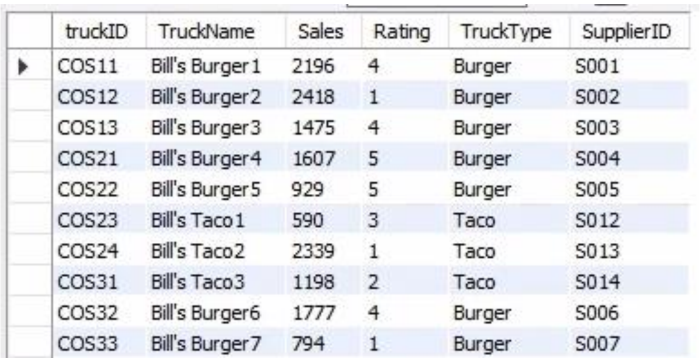
locationID	nooflots
A1	3
A2	6
A3	10
A4	8
A5	4
B1	4
B2	5
B3	4
B4	7
B5	7

h. **supplier**



	SupplierID	Location	SupplierType	SupplierName
▶	S001	Livingston Oval	Burgers	Burger Supplier 1
	S002	Hasbrouck Heights	Burgers	Burger Supplier 11
	S003	Hasbrouck Heights	Burgers	Sub Supplier 3
	S004	Hasbrouck Heights	Burgers	Sub Supplier 8
	S005	Kearny Avenue	Burgers	Burger Supplier 6
	S006	Kearny Avenue	Burgers	Taco Supplier 10
	S007	Maywood	Burgers	Taco Supplier 1
	S008	Maywood	Burgers	Taco Supplier 2
	S009	Maywood	Burgers	Sub Supplier 4
	S010	Kearny Avenue	Burgers	Burger Supplier 2

i. **foodtruck**



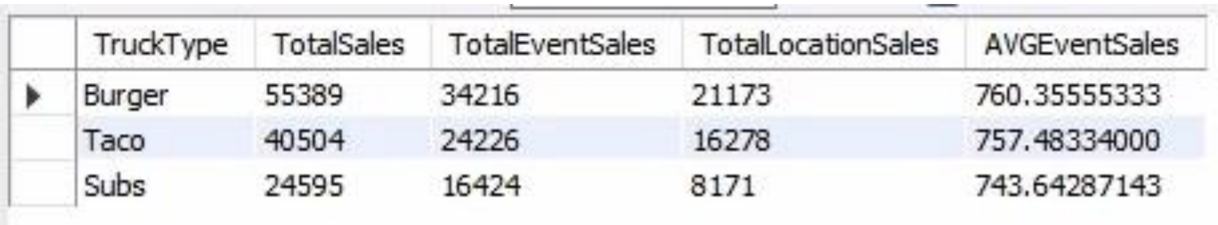
	truckID	TruckName	Sales	Rating	TruckType	SupplierID
▶	COS11	Bill's Burger1	2196	4	Burger	S001
	COS12	Bill's Burger2	2418	1	Burger	S002
	COS13	Bill's Burger3	1475	4	Burger	S003
	COS21	Bill's Burger4	1607	5	Burger	S004
	COS22	Bill's Burger5	929	5	Burger	S005
	COS23	Bill's Taco1	590	3	Taco	S012
	COS24	Bill's Taco2	2339	1	Taco	S013
	COS31	Bill's Taco3	1198	2	Taco	S014
	COS32	Bill's Burger6	1777	4	Burger	S006
	COS33	Bill's Burger7	794	1	Burger	S007

VII. SQL Queries

1. After finding the most profitable truck types combined with their average ratings, Purdue Mobile Food Solutions (PMFS) hopes to purchase more of the most popular and efficient truck types in the future so that budgets will be spent more efficiently in business expansion. This query gives PMFS an overview of each TruckType's sales by location, event, average event sales, total sales and average rating.

   RIGHT JOIN and SUBQUERY :
   List the total sales (Event Sales and Location sales) for each truck type in descending order and provide the average rating for each truck type

```
SELECT foodtruck. TruckType, SUM(foodtruck.Sales+TotalEventSales) AS TotalSales,

SUM(TotalEventSales) AS TotalEventSales, SUM(Sales) AS TotalLocationSales,

AVG(AVGEventSales) AS AVGEventSales

FROM foodtruck

RIGHT JOIN(

SELECT SUM(EventSales) as TotalEventSales, truckID, COUNT(*), AVG(EventSales)

AS AVGEventSales

FROM foodtruck_event

GROUP BY truckID) A

ON foodtruck.truckID = A.truckID

GROUP BY foodtruck.TruckType

ORDER BY  foodtruck.Sales+TotalEventSales DESC;
```



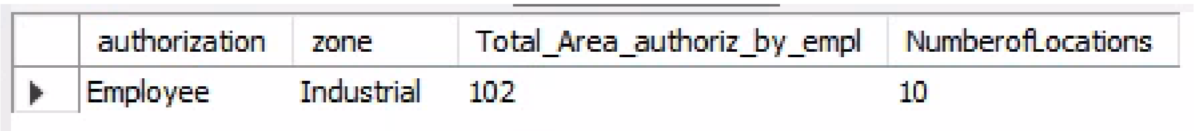
	TruckType	TotalSales	TotalEventSales	TotalLocationSales	AVGEventSales
▶	Burger	55389	34216	21173	760.35555333
	Taco	40504	24226	16278	757.48334000
	Subs	24595	16424	8171	743.64287143

2. The CFO of PMFS wants to figure out the overall area size and number of locations in which employee authorization is required so that they can plan their business expansion

and budget more carefully since such authorization is acquired through negotiations and time-consuming.

GROUP BY:
List the total area size and count for zones in which employee authorization is required.

```
SELECT authorization, zone, SUM(areaSize) as Total_Area_authoriz_by_empl,
COUNT(*) AS NumberofLocations

FROM location

WHERE authorization = "employee"

GROUP BY zone;
```



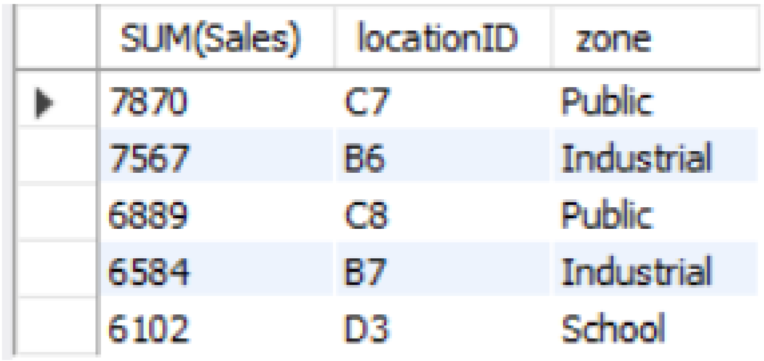
	authorization	zone	Total_Area_authoriz_by_empl	NumberofLocations
▶	Employee	Industrial	102	10

3. In order to save gasoline costs and make investments transit into profits more efficiently, PMFS wants to target some "golden" locations to deploy food trucks in search of more visits, reputation, and sales. After this query provides them with the top 5 most profitable locations, PMFS can check and ensure they always have a truck at those locations to maximise profit. Other than that, PMFS can also learn which zones are associated with the high sales locations and research to more locations under the same zone tagging to deploy foodtrucks.

INNER JOIN AND GROUP BY :
List five most profitable locations to deploy food trucks. Display their sales, location and zone.

```
SELECT SUM(Sales),location_foodtruck.locationID,location.zone

FROM foodtruck

INNER JOIN location_foodtruck

ON foodtruck.truckID = location_foodtruck.truckID

INNER JOIN location

ON location_foodtruck.locationID = location.locationID

GROUP BY locationID

ORDER BY sum(Sales) DESC
```

```
LIMIT 5;
```

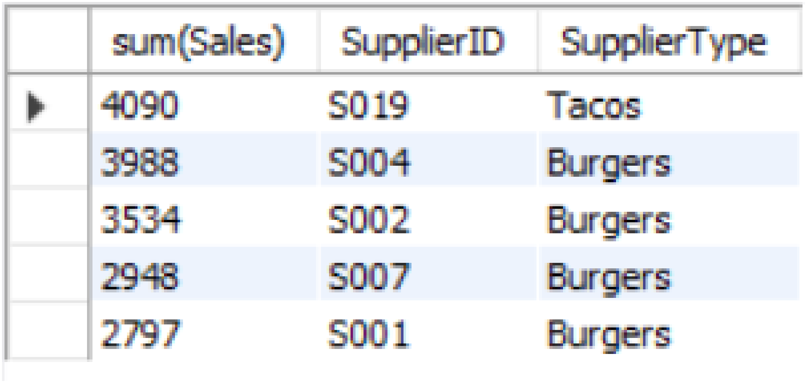


SUM(Sales)	locationID	zone
7870	C7	Public
7567	B6	Industrial
6889	C8	Public
6584	B7	Industrial
6102	D3	School

4. Suppliers of PMFS vary in type and scale. It is time to know what customers like most. With the analyses, PMFS can cut its budgets in purchasing less popular food and focus on cooperating with the most popular and profitable suppliers.

LEFT JOIN and GROUP BY :
List five suppliers whose food was the most popular.

```
SELECT sum(Sales),supplier.SupplierID,supplier.SupplierType

FROM foodtruck

LEFT JOIN supplier

ON foodtruck.SupplierID = supplier.SupplierID

GROUP BY SupplierID

ORDER BY sum(Sales) DESC

LIMIT 5;
```

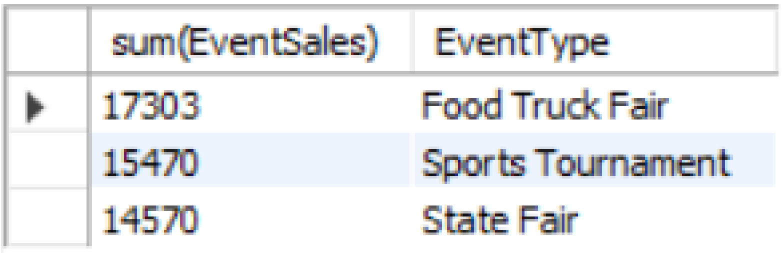


sum(Sales)	SupplierID	SupplierType
4090	S019	Tacos
3988	S004	Burgers
3534	S002	Burgers
2948	S007	Burgers
2797	S001	Burgers

5. Events are great places to increase the popularity and sales of PMFS. However, there have been so many events taking place in this city. PMFS does not have enough employees and budgets to attend all of them. It is strongly recommended that PMFS target the three most profitable event types and gain reputation and sales from them.

INNER JOIN and GROUP BY:
List Top 3 event types that incurs the most sales.

```
SELECT sum(EventSales),EventType

FROM foodtruck

INNER JOIN foodtruck_event

ON foodtruck.truckID = foodtruck_event.TruckID

INNER JOIN event

ON event.EventID = foodtruck_event.EventID

GROUP BY event.EventType

ORDER BY sum(EventSales) DESC

LIMIT 3;
```



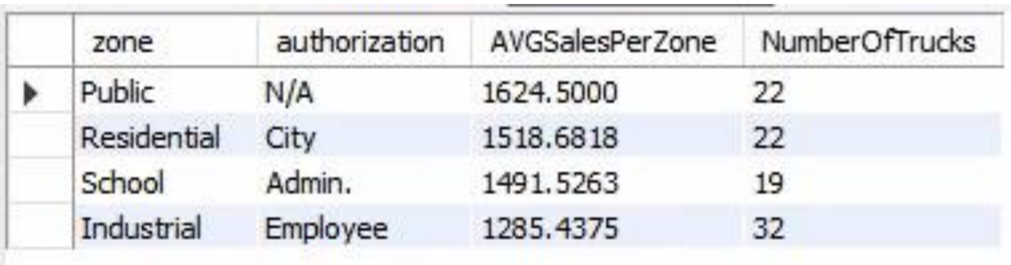
	sum(EventSales)	EventType
▶	17303	Food Truck Fair
	15470	Sports Tournament
	14570	State Fair

6. To plan budgets and earn profits better, PMFS wants to know which zone has the highest average sales per truck and then compares it to the number of trucks it currently has in that zone to determine if PMFS needs to rearrange the current truck positioning.

INNER JOIN and GROUP BY :
List every zones average sales and the associated number of trucks

```
SELECT  location.zone,location.authorization, AVG(foodtruck.sales) AS
AVGSalesPerZone, COUNT(foodtruck.truckID) AS NumberOfTrucks

FROM  location
```

```
INNER JOIN location_foodtruck

ON location.locationID = location_foodtruck.locationID

INNER JOIN foodtruck

ON location_foodtruck.truckID = foodtruck.truckID

GROUP BY location.zone,location.authorization

ORDER BY AVG(foodtruck.sales) DESC;
```

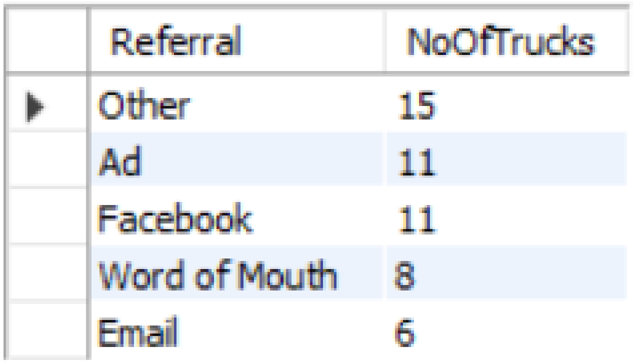


	zone	authorization	AVGSalesPerZone	NumberOfTrucks
▶	Public	N/A	1624.5000	22
	Residential	City	1518.6818	22
	School	Admin.	1491.5263	19
	Industrial	Employee	1285.4375	32

7. Advertisements have been a great method in expanding business and gaining reputation. After knowing the most successful referral methods, PMFS can invest more into that type while cutting budgets on other inefficient referral methods to attract trucks. Considering the high number of "Other" referrals, PMFS can also look into other types of advertising that they have not actively engaged in but is driving high volume.

LEFT JOIN and GROUP BY:
List the total number of trucks associated with each referral

```
SELECT location_foodtruck.Referral,COUNT(foodtruck.truckID) AS NoOfTrucks

FROM foodtruck

LEFT JOIN location_foodtruck

ON foodtruck.truckID = location_foodtruck.truckID

WHERE foodtruck.TruckType = "Burger"

GROUP BY location_foodtruck.Referral

ORDER BY COUNT(foodtruck.truckID) DESC;
```



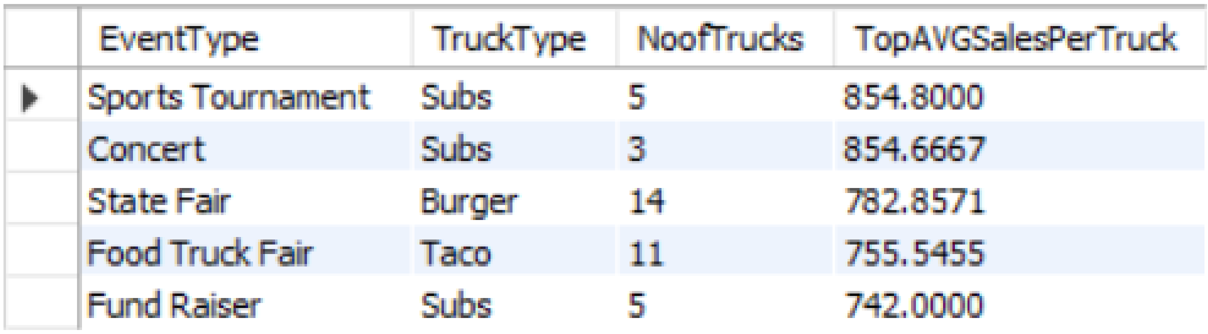
Referral	NoOfTrucks
Other	15
Ad	11
Facebook	11
Word of Mouth	8
Email	6

8. PMFS wants to know which truck type performs best in attracting customers and promoting sales during each event so that it can plan more specifically and deploy that truck type more in the incoming events.

INNER JOIN and SUBQUERY:
List the truck type that has the highest average sales for each event type.

```
SELECT EventType, TruckType, NoofTrucks, Max(AVGTruckSales) AS
TopAVGSalesPerTruck

FROM (

SELECT event.EventType, foodtruck.TruckType,

        COUNT(foodtruck.TruckID) AS NoofTrucks,

        AVG(foodtruck_event.EventSales) AS AVGTruckSales

        FROM event

        INNER JOIN foodtruck_event

        ON event.EventID = foodtruck_event.EventID

        INNER JOIN foodtruck

        ON foodtruck.TruckID = foodtruck_event.TruckID

        GROUP BY event.EventType, foodtruck.TruckType

        ORDER BY event.EventType DESC, AVG(foodtruck_event.EventSales)

        DESC

) A
```

```
GROUP BY EventType

ORDER BY TopAVGSalesPerTruck DESC;
```

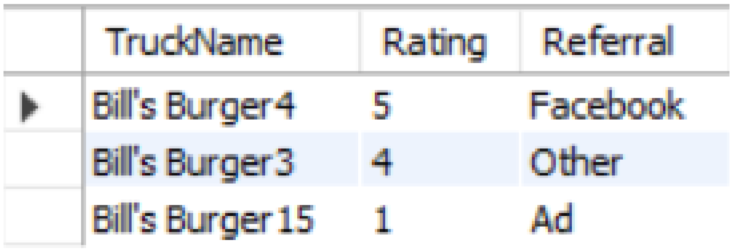


	EventType	TruckType	NoofTrucks	TopAVGSalesPerTruck
▶	Sports Tournament	Subs	5	854.8000
	Concert	Subs	3	854.6667
	State Fair	Burger	14	782.8571
	Food Truck Fair	Taco	11	755.5455
	Fund Raiser	Subs	5	742.0000

9. PMFS has received several calls that praised its service in Location B1. Therefore, we want to award employees in this area according to their contributions to sales, which can act as an encouragement to other employees.

RIGHT JOIN:
List the top 3 highest rated food trucks at Location B1 with the corresponding method of referral.

```
SELECT foodtruck.TruckName, foodtruck.Rating, location_foodtruck.Referral

FROM foodtruck

RIGHT JOIN location_foodtruck

ON foodtruck.truckID = location_foodtruck.truckID

WHERE location_foodtruck.locationID="B1"

ORDER BY Rating DESC

LIMIT 3;
```

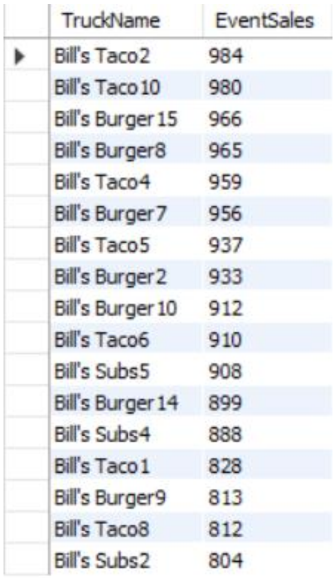


	TruckName	Rating	Referral
▶	Bill's Burger4	5	Facebook
	Bill's Burger3	4	Other
	Bill's Burger15	1	Ad

10. It is necessary that PMFS boost sales by encouraging its employees and awarding them accordingly. But first of all, it needs to figure out the sales champions during these months.

LEFT JOIN and GROUP BY:
List all the food trucks which have event sales greater than $800.

```
SELECT foodtruck.TruckName, foodtruck_event.EventSales

FROM foodtruck

LEFT JOIN foodtruck_event

ON foodtruck.truckID = foodtruck_event.TruckID

GROUP BY foodtruck.TruckName

HAVING foodtruck_event.EventSales>800

ORDER BY foodtruck_event.EventSales DESC;
```

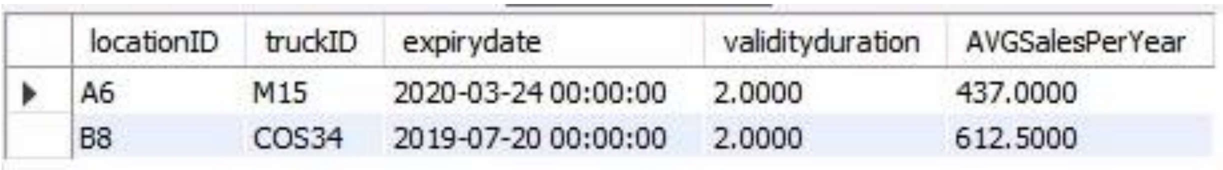


	TruckName	EventSales
▶	Bill's Taco2	984
	Bill's Taco10	980
	Bill's Burger15	966
	Bill's Burger8	965
	Bill's Taco4	959
	Bill's Burger7	956
	Bill's Taco5	937
	Bill's Burger2	933
	Bill's Burger10	912
	Bill's Taco6	910
	Bill's Subs5	908
	Bill's Burger14	899
	Bill's Subs4	888
	Bill's Taco1	828
	Bill's Burger9	813
	Bill's Taco8	812
	Bill's Subs2	804

11. PMFS realized that there was a system error a few months ago that caused their franchise officer to not receive notifications on their expired licenses. PMFS now needs to find out which licenses has expired and apply for a renewal as soon as possible to avoid any penalty from the authorities. Before renewing any licenses, PMFS also studies the trucks performance at the specific location to determine if the license is worth renewing or not.

RIGHT JOIN and SUBQUERY:
List the franchise licenses that are currently expired. Also display their validity duration and average sales per valid year.

```
SELECT A.*, foodtruck.Sales/validityduration AS AVGSalesPerYear

FROM foodtruck

RIGHT JOIN

(SELECT locationID, truckID, expirydate,

DATEDIFF(franchiselicense.expirydate,franchiselicense.dateissued)/365 AS

validityduration

FROM franchiselicense) A

ON foodtruck.truckID = A.truckID

WHERE expirydate<CURDATE();
```



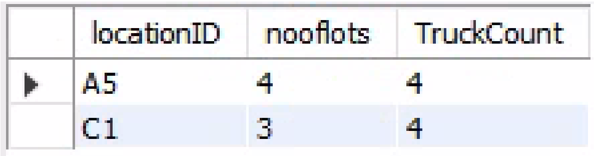
	locationID	truckID	expirydate	validityduration	AVGSalesPerYear
▶	A6	M15	2020-03-24 00:00:00	2.0000	437.0000
	B8	COS34	2019-07-20 00:00:00	2.0000	612.5000

12. Many of the foodtrucks under PMFS has deploying a lot of foodtrucks are different locations. After the recent tightening of number of lots per permit location, PMFS needs to find out if they have too many trucks at any locations or if they're currently at the max number allowed so that they stop deploying any foodtrucks at that locations.

INNER JOIN and SUBQUERY:
List the locations which has a total truck count that's more than or equal to the permitted no of lots.

```
SELECT location_permit.locationID, location_permit.nooflots, TruckCount

FROM location_permit

INNER JOIN(

SELECT COUNT(*) AS TruckCount, locationID

FROM location_foodtruck

GROUP BY locationID) A
```

```
ON location_permit.locationID = A.locationID

WHERE location_permit.nooflots < TruckCount OR location_permit.nooflots =
TruckCount;
```

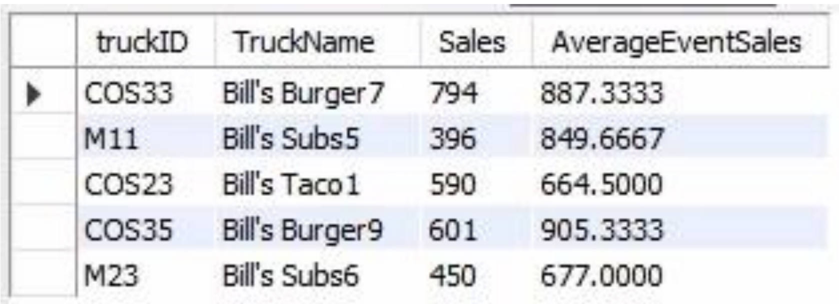


	locationID	nooflots	TruckCount
▶	A5	4	4
	C1	3	4

13. During a recent performance review after an event, PMFS noticed that there were trucks that usually have low sales amount generating much higher event sales. PMFS wants to find out which trucks share this trait and redirect these trucks operating time to cater events more than location based sales.

RIGHT JOIN and SUBQUERY:
List trucks that has an average event sales that’s higher than their regular sales

```
SELECT foodtruck.truckID, foodtruck.TruckName, foodtruck.Sales, AverageEventSales

FROM foodtruck

RIGHT JOIN

(SELECT AVG(foodtruck_event.EventSales) AS AverageEventSales,
foodtruck_event.truckID

FROM foodtruck_event

GROUP BY foodtruck_event.TruckID

ORDER BY foodtruck_event.EventSales DESC)A

ON A.truckID = foodtruck.truckID

WHERE foodtruck.Sales < AverageEventSales;
```

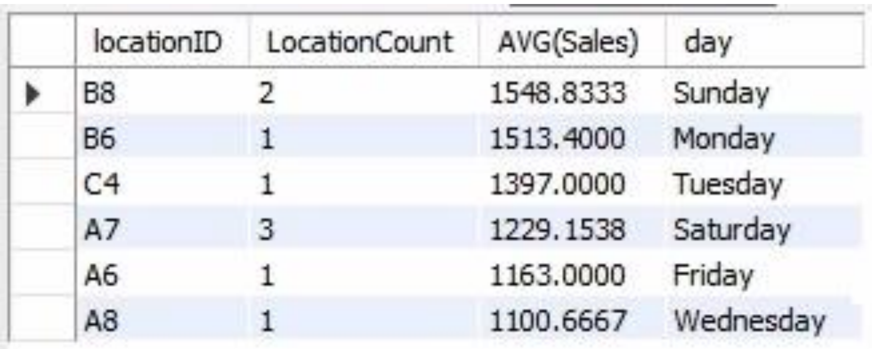


	truckID	TruckName	Sales	AverageEventSales
▶	COS33	Bill's Burger7	794	887.3333
	M11	Bill's Subs5	396	849.6667
	COS23	Bill's Taco1	590	664.5000
	COS35	Bill's Burger9	601	905.3333
	M23	Bill's Subs6	450	677.0000

14. Franchise permits are expensive and PMFS needs to decide ways to better decide which licenses to obtain. PMFS decides that they want to study if different days of the week affects the average sales amount per truck.

LEFT JOIN and GROUP BY:
List the average sales amount for the days that franchise locations are open.

```
SELECT location_franchise.locationID,

Count(DISTINCT(location_franchise.locationID)) AS LocationCount,AVG(Sales), day

FROM location_franchise

LEFT JOIN location_foodtruck

ON location_franchise.locationID = location_foodtruck.locationID

LEFT JOIN foodtruck

ON location_foodtruck.truckID = foodtruck.truckID

GROUP BY day

ORDER BY AVG(Sales)

DESC;
```

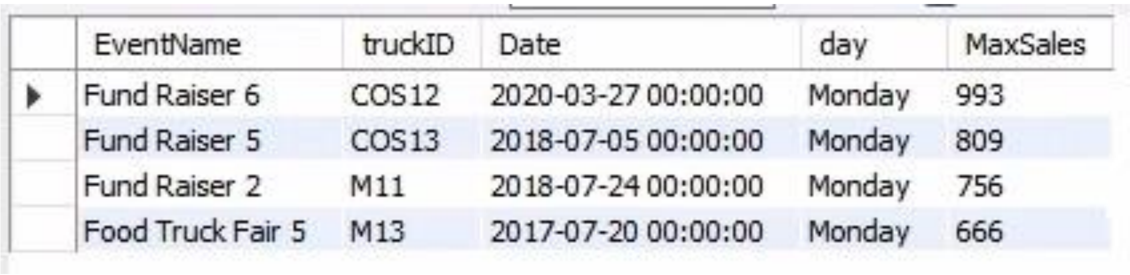


	locationID	LocationCount	AVG(Sales)	day
▶	B8	2	1548.8333	Sunday
	B6	1	1513.4000	Monday
	C4	1	1397.0000	Tuesday
	A7	3	1229.1538	Saturday
	A6	1	1163.0000	Friday
	A8	1	1100.6667	Wednesday

15. The foodtruck manager of PMFS is trying to remember which truck had the highest sales during a particular event. He only knows that the event was held on Monday and that the name started with F. He believes he can determine which truck it was if he had a list of all the highest sale amount from events that suits his description.

LEFT JOIN and SUBQUERY:
List the Max event sales for any event that starts with F and occurred on Monday. Display this information with the truckID.

```
SELECT EventName,truckID, Date, day, MaxSales

FROM event

LEFT JOIN

(SELECT MAX(EventSales) AS MaxSales, eventID, truckID

FROM foodtruck_event

GROUP BY eventID)A

ON event.eventID = A.eventID

WHERE EventName LIKE "F%" AND Day="Monday"

ORDER BY MaxSales

  DESC
```



EventName	truckID	Date	day	MaxSales
Fund Raiser 6	COS12	2020-03-27 00:00:00	Monday	993
Fund Raiser 5	COS13	2018-07-05 00:00:00	Monday	809
Fund Raiser 2	M11	2018-07-24 00:00:00	Monday	756
Food Truck Fair 5	M13	2017-07-20 00:00:00	Monday	666