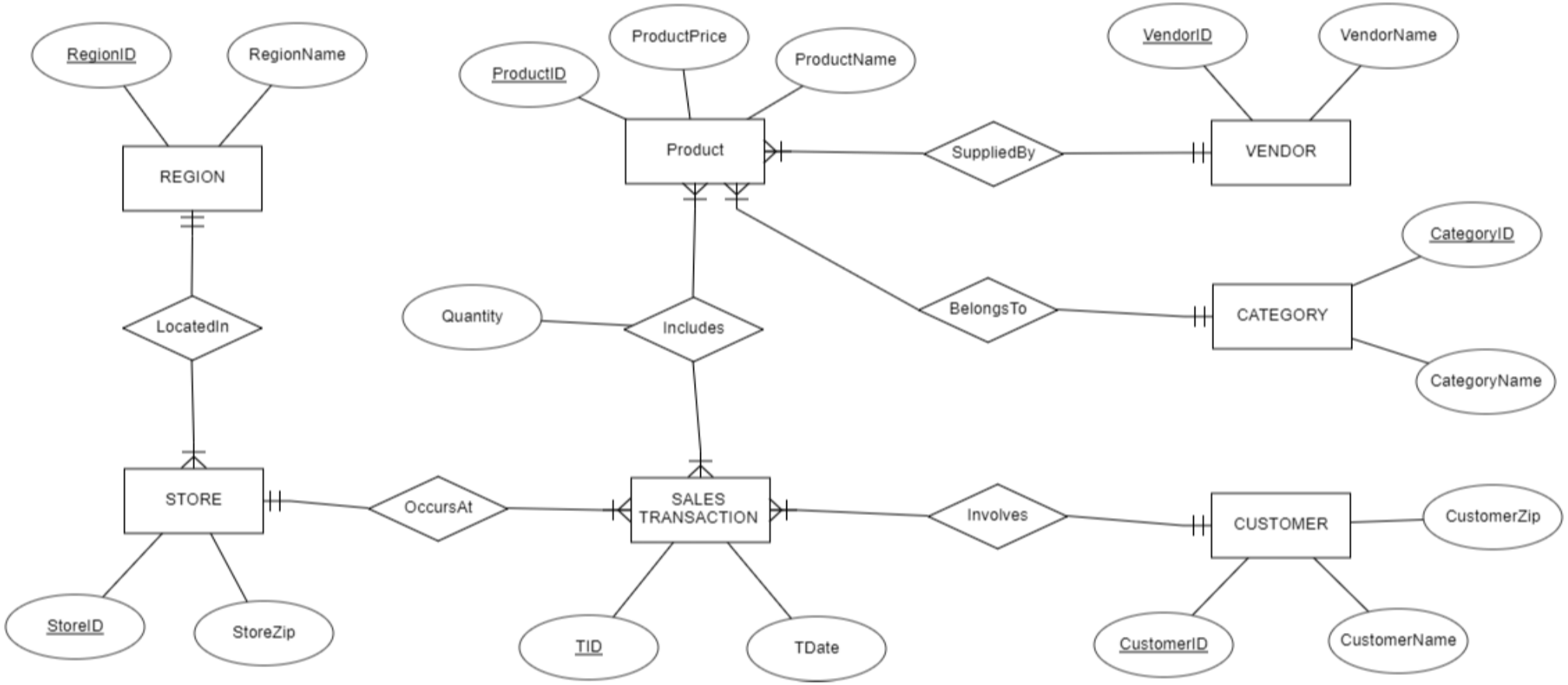
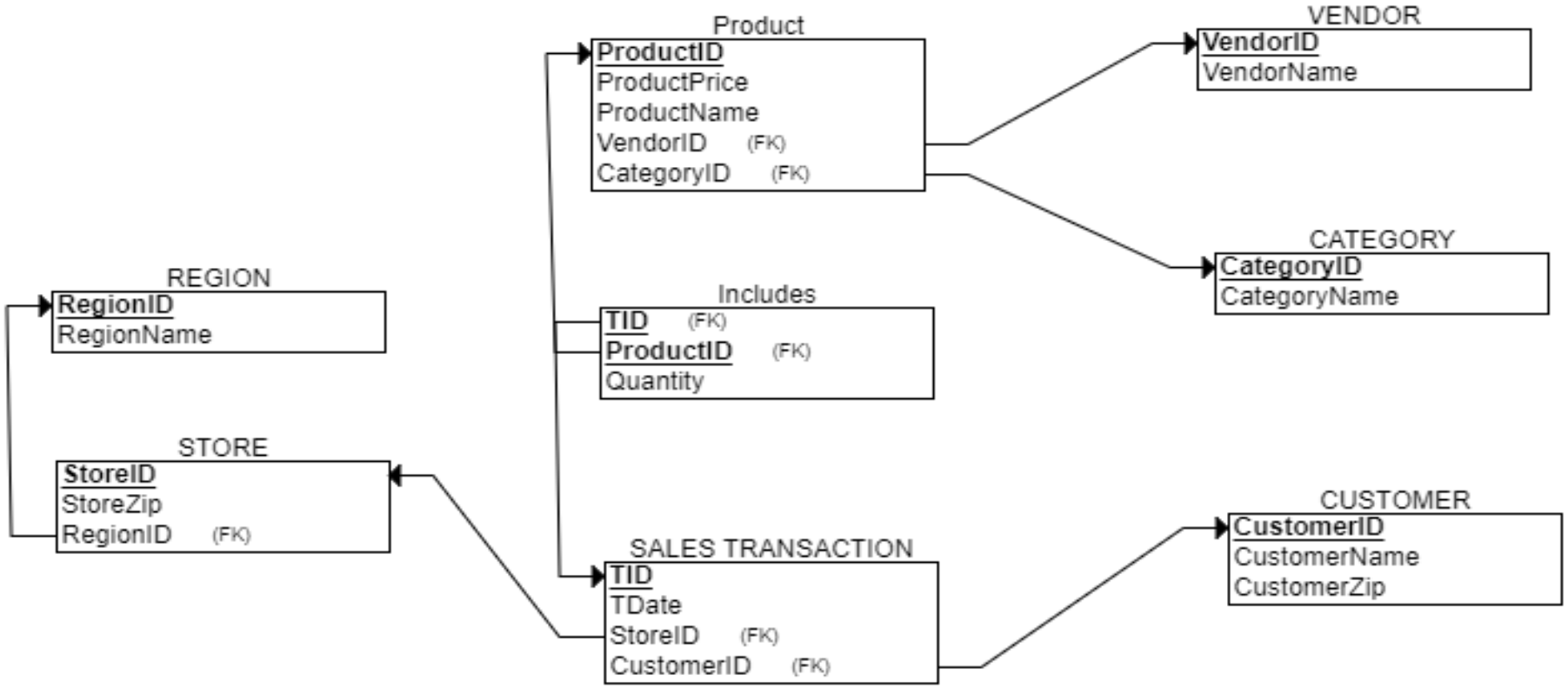
1. ZAGI RETAIL COMPANY – ER DIAGRAM



2. RELATIONAL SCHEMA



3. DATA DICTIONARY

Entity: **Vendor** – This table contains information about the vendors who provide products

Field Name	Description	Type	Specifications	Required	Unique	Key(s)
vendorid	Vendor ID	CHAR	2 numeric characters	Yes	Yes	PK
vendorname	Vendor name	VARCHAR	25 alpha-numeric characters	Yes	No	

Entity: **Category** – This table contains information about the categories of products

Field Name	Description	Type	Specifications	Required	Unique	Key(s)
categoryid	Category ID	CHAR	2 numeric characters	Yes	Yes	PK
categoryname	Category name	VARCHAR	25 alpha-numeric characters	Yes	No	

3. DATA DICTIONARY

Entity: **Product** – This table contains information about the products

Field Name	Description	Type	Specifications	Required	Unique	Key(s)
productid	Product ID	CHAR	3 numeric characters	Yes	Yes	PK
productname	Product name	VARCHAR	25 alpha-numeric characters	Yes	No	
productprice	Product price	NUMERIC	7 digits with 2 decimals numeric	Yes	No	
vendorid	Vendor ID	CHAR	2 numeric characters	Yes	No	FK Vendor
categoryid	Category ID	CHAR	2 numeric characters	Yes	No	FK Category

Entity: **Region** – This table contains information about the store's regions

Field Name	Description	Type	Specifications	Required	Unique	Key(s)
regionid	Region ID	CHAR	1 numeric character	Yes	Yes	PK
regionname	Region name	VARCHAR	25 alpha-numeric characters	Yes	No	

3. DATA DICTIONARY

Entity: **Store** – This table contains information about the stores

Field Name	Description	Type	Specifications	Required	Unique	Key(s)
storeid	Store ID	VARCHAR	3 alpha-numeric characters	Yes	Yes	PK
storezip	Store Zip code	CHAR	5 numeric characters	Yes	No	
regionid	Region ID	CHAR	1 numeric character	Yes	No	FK Region

Entity: **Customer** – This table contains information about the customers who purchase products

Field Name	Description	Type	Specifications	Required	Unique	Key(s)
customerid	Customer ID	CHAR	7 numeric characters	Yes	Yes	PK
customername	Customer name	VARCHAR	15 alpha-numeric characters	Yes	No	
customerzip	Customer Zip number	CHAR	5 numeric characters	Yes	No	

3. DATA DICTIONARY

Entity: **Salestransaction** – This table contains information about the sales transactions

Field Name	Description	Type	Specifications	Required	Unique	Key(s)
tid	Transaction ID	VARCHAR	8 alpha-numeric characters	Yes	Yes	PK
customerid	Customer ID	CHAR	7 numeric characters	Yes	No	FK Customer
storeid	Store ID	VARCHAR	3 alpha-numeric characters	Yes	No	FK Store
tdate	Transaction Date	DATE	MM/DD/YYYY format	Yes	No	

Entity: **Includes** – This table contains information about the related sales transactions.

Field Name	Description	Type	Specifications	Required	Unique	Key(s)
productid	Product ID	CHAR	3 numeric characters	Yes	Yes	PK, FK Product
tid	Transaction ID	VARCHAR	8 alpha-numeric characters	Yes	Yes	PK, FK Salestransaction
quantity	Quantity	INT	1 digit; quantity count	Yes	No	

4. RELATIONAL DATABASE CREATION – SQL

```
CREATE TABLE vendor(
 vendorid CHAR(2) NOT NULL,
 vendorname VARCHAR(25) NOT NULL,
 PRIMARY KEY(vendorid)
 );

CREATE TABLE category(
 categoryid CHAR(2) NOT NULL,
 categoryname VARCHAR(25) NOT NULL,
 PRIMARY KEY(categoryid)
 );

CREATE TABLE product(
 productid CHAR(3) NOT NULL,
 productname VARCHAR(25) NOT NULL,
 productprice NUMERIC(7, 2) NOT NULL,
 vendorid CHAR(2) NOT NULL,
 categoryid CHAR(2) NOT NULL,
 PRIMARY KEY(productid),
 FOREIGN KEY(vendorid) REFERENCES vendor(vendorid),
 FOREIGN KEY(categoryid) REFERENCES category(categoryid)
 );

CREATE TABLE region(
 regionid CHAR(1) NOT NULL,
 regionname VARCHAR(25) NOT NULL,
 PRIMARY KEY(regionid)
 );

CREATE TABLE store(
 storeid VARCHAR(3) NOT NULL,
 storezip CHAR(5) NOT NULL,
 regionid CHAR(1) NOT NULL,
 PRIMARY KEY(storeid),
 FOREIGN KEY(regionid) REFERENCES region(regionid)
 );

CREATE TABLE customer(
 customerid CHAR(7) NOT NULL,
 customername VARCHAR(15) NOT NULL,
 customerzip CHAR(5) NOT NULL,
 PRIMARY KEY(customerid)
 );

CREATE TABLE salestransaction(
 tid VARCHAR(8) NOT NULL,
 customerid CHAR(7) NOT NULL,
 storeid VARCHAR(3) NOT NULL,
 tdate DATE NOT NULL,
 PRIMARY KEY(tid),
 FOREIGN KEY(customerid) REFERENCES customer(customerid),
 FOREIGN KEY(storeid) REFERENCES store(storeid)
 );

CREATE TABLE includes(
 productid CHAR(3) NOT NULL,
 tid VARCHAR(8) NOT NULL,
 quantity INT NOT NULL,
 PRIMARY KEY(productid, tid),
 FOREIGN KEY(productid) REFERENCES product(productid),
 FOREIGN KEY(tid) REFERENCES salestransaction(tid)
 );
```

5. POPULATING THE DATABASE – SQL

```
INSERT INTO vendor VALUES ('PG','Pacifica Gear');
INSERT INTO vendor VALUES ('MK','Mountain King');
INSERT INTO category VALUES ('CP','Camping');
INSERT INTO category VALUES ('FW','Footwear');
INSERT INTO product VALUES ('1X1','Zzz Bag',100,'PG','CP');
INSERT INTO product VALUES ('2X2','Easy Boot',70,'MK','FW');
INSERT INTO product VALUES ('3X3','Cosy Sock',15,'MK','FW');
INSERT INTO product VALUES ('4X4','Dura Boot',90,'PG','FW');
INSERT INTO product VALUES ('5X5','Tiny Tent',150,'MK','CP');
INSERT INTO product VALUES ('6X6','Biggy Tent',250,'MK','CP');
INSERT INTO region VALUES ('C','Chicagoland');
INSERT INTO region VALUES ('T','Tristate');
INSERT INTO store VALUES ('S1','60600','C');
INSERT INTO store VALUES ('S2','60605','C');
INSERT INTO store VALUES ('S3','35400','T');
INSERT INTO customer VALUES ('1-2-333','Tina','60137');
INSERT INTO customer VALUES ('2-3-444','Tony','60611');
INSERT INTO customer VALUES ('3-4-555','Pam','35401');
INSERT INTO salestransaction VALUES ('T111','1-2-333','S1','2020-01-01');
INSERT INTO salestransaction VALUES ('T222','2-3-444','S2','2020-01-01');
INSERT INTO salestransaction VALUES ('T333','1-2-333','S3','2020-01-02');
INSERT INTO salestransaction VALUES ('T444','3-4-555','S3','2020-01-02');
INSERT INTO salestransaction VALUES ('T555','2-3-444','S3','2020-01-02');
INSERT INTO includes VALUES ('1X1','T111',1);
INSERT INTO includes VALUES ('2X2','T222',1);
INSERT INTO includes VALUES ('3X3','T333',5);
INSERT INTO includes VALUES ('1X1','T333',1);
INSERT INTO includes VALUES ('4X4','T444',1);
INSERT INTO includes VALUES ('2X2','T444',2);
INSERT INTO includes VALUES ('4X4','T555',4);
INSERT INTO includes VALUES ('5X5','T555',2);
INSERT INTO includes VALUES ('6X6','T555',1);
```

6. SQL QUERIES

WHERE

Display all the information for all stores whose RegionID value is C.

```
SELECT *
FROM store
WHERE regionid='c';
```

storeid	storezip	regionid
S1	60600	C
S2	60605	C
NULL	NULL	NULL

ORDER BY

Display the CustomerName and CustomerZip for all customers, sorted alphabetically by CustomerName.

```
SELECT customername, customerzip
FROM customer
ORDER BY customername;
```

customername	customerzip
Pam	35401
Tina	60137
Tony	60611

6. SQL QUERIES

JOIN

Display the ProductID, ProductName, ProductPrice, and VendorName for all products. Sort the results by ProductID.

```
SELECT productid, productname, productprice, vendorname
FROM product
JOIN vendor
ON product.vendorid=vendor.vendorid
ORDER BY productid;
```

productid	productname	productprice	vendorname
1X1	Zzz Bag	100.00	Pacifica Gear
2X2	Easy Boot	70.00	Mountain King
3X3	Cosy Sock	15.00	Mountain King
4X4	Dura Boot	90.00	Pacifica Gear
5X5	Tiny Tent	150.00	Mountain King
6X6	Biggy Tent	250.00	Mountain King

LEFT OUTER JOIN

Display the ProductID, ProductName, ProductPrice, VendorName, and CategoryName for all products. Sort the results by ProductID.

```
SELECT productid, productname, productprice, vendorname, categoryname
FROM product
LEFT OUTER JOIN vendor
ON product.vendorid=vendor.vendorid
LEFT OUTER JOIN category
ON product.categoryid=category.categoryid
ORDER BY productid;
```

productid	productname	productprice	vendorname	categoryname
1X1	Zzz Bag	100.00	Pacifica Gear	Camping
2X2	Easy Boot	70.00	Mountain King	Footwear
3X3	Cosy Sock	15.00	Mountain King	Footwear
4X4	Dura Boot	90.00	Pacifica Gear	Footwear
5X5	Tiny Tent	150.00	Mountain King	Camping
6X6	Biggy Tent	250.00	Mountain King	Camping

6. SQL QUERIES

HAVING

Display the TID and the total number of items (of all products) sold within the transaction for all sales transactions whose total number of items (of all products) sold within the transaction is greater than five.

```
SELECT tid, SUM(quantity)
FROM includes
GROUP BY tid
HAVING SUM(quantity)>5;
```

tid	SUM(quantity)
T333	6
T555	7

6. SQL QUERIES

Set operations (UNION, INTERSECT, EXCEPT)

Display the ProductID, ProductName, ProductPrice for each product that has more than two items sold within all sales transactions.

```
CREATE VIEW products_sold_3 AS
SELECT productid, productname, productprice
FROM product
WHERE productid IN (SELECT productid FROM includes GROUP BY productid HAVING SUM(quantity)>2);

CREATE VIEW products_trans_2 AS
SELECT productid, productname, productprice
FROM product
WHERE productid IN (SELECT productid FROM includes GROUP BY productid);

SELECT *
FROM products_sold_3
UNION
SELECT *
FROM products_trans_2;
```

productid	productname	productprice
3X3	Cosy Sock	15.00
4X4	Dura Boot	90.00
1X1	Zzz Bag	100.00
2X2	Easy Boot	70.00
5X5	Tiny Tent	150.00
6X6	Biggy Tent	250.00

6. SQL QUERIES

Subqueries

Display the ProductID, ProductName, ProductPrice and VendorName for products whose price is below the average price of all products.

```
SELECT productid, productname, productprice, vendorname
FROM product
JOIN vendor
ON vendor.vendorid=product.vendorid
WHERE productprice<(SELECT AVG(productprice) FROM product);
```

	productid	productname	productprice	vendorname
▶	3X3	Cosy Sock	15.00	Mountain King
	2X2	Easy Boot	70.00	Mountain King
	4X4	Dura Boot	90.00	Pacifica Gear
	1X1	Zzz Bag	100.00	Pacifica Gear

Aggregate functions

Display the ProductID, ProductName, and ProductPrice of the cheapest product.

```
SELECT productid, productname, min(productprice)
FROM product
GROUP BY productid;
```

	productid	productname	min(productprice)
▶	1X1	Zzz Bag	15.00

6. SQL QUERIES

Database updates using INSERT, UPDATE and DELETE

```
INSERT INTO customer VALUES ('4-5-666','John','95819');
```

customerid	customername	customerzip
1-2-333	Tina	60137
2-3-444	Tony	60611
3-4-555	Pam	35401
4-5-666	John	95819
NULL	NULL	NULL

```
UPDATE customer
SET customername='Jane'
WHERE customername='John';
```

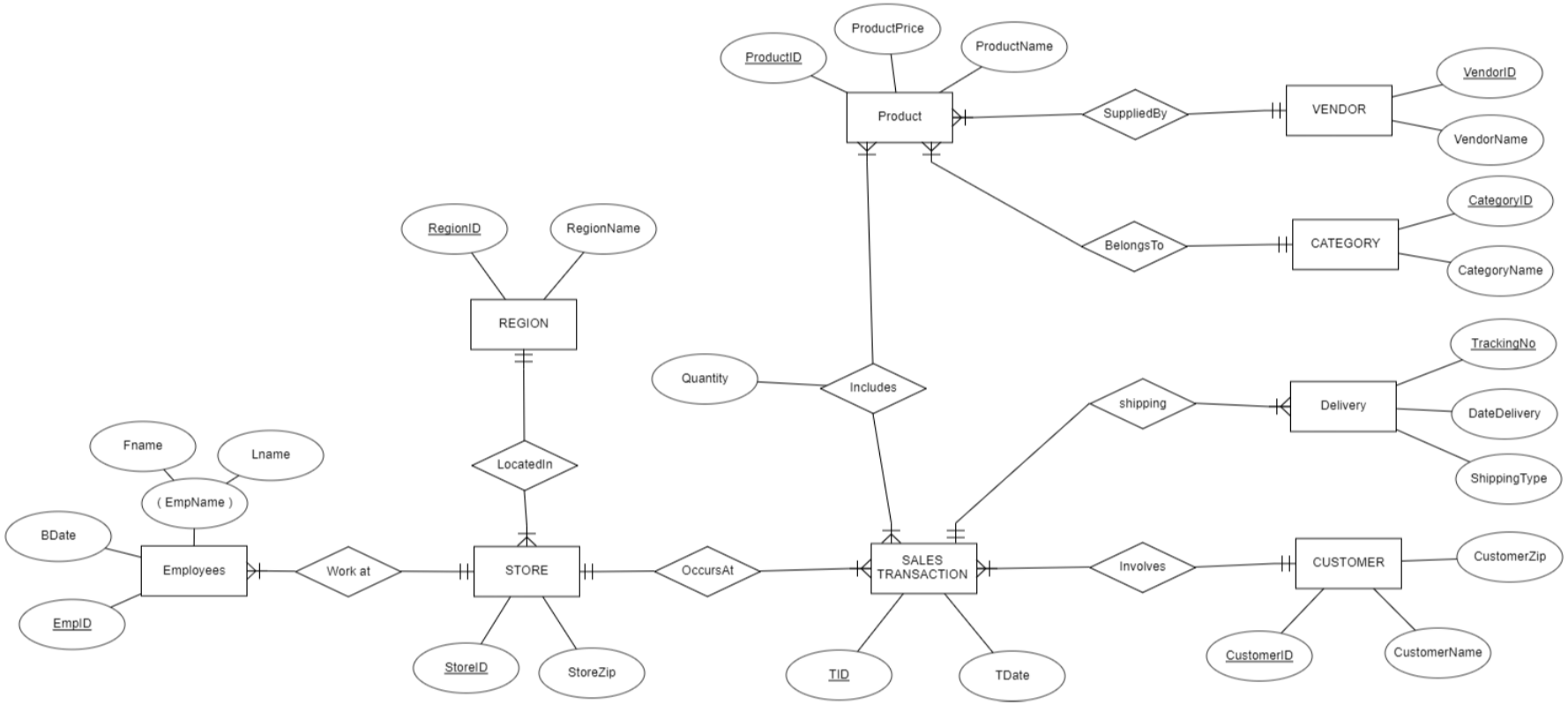
customerid	customername	customerzip
1-2-333	Tina	60137
2-3-444	Tony	60611
3-4-555	Pam	35401
4-5-666	Jane	95819
NULL	NULL	NULL

```
DELETE FROM customer
WHERE customername='Jane';
```

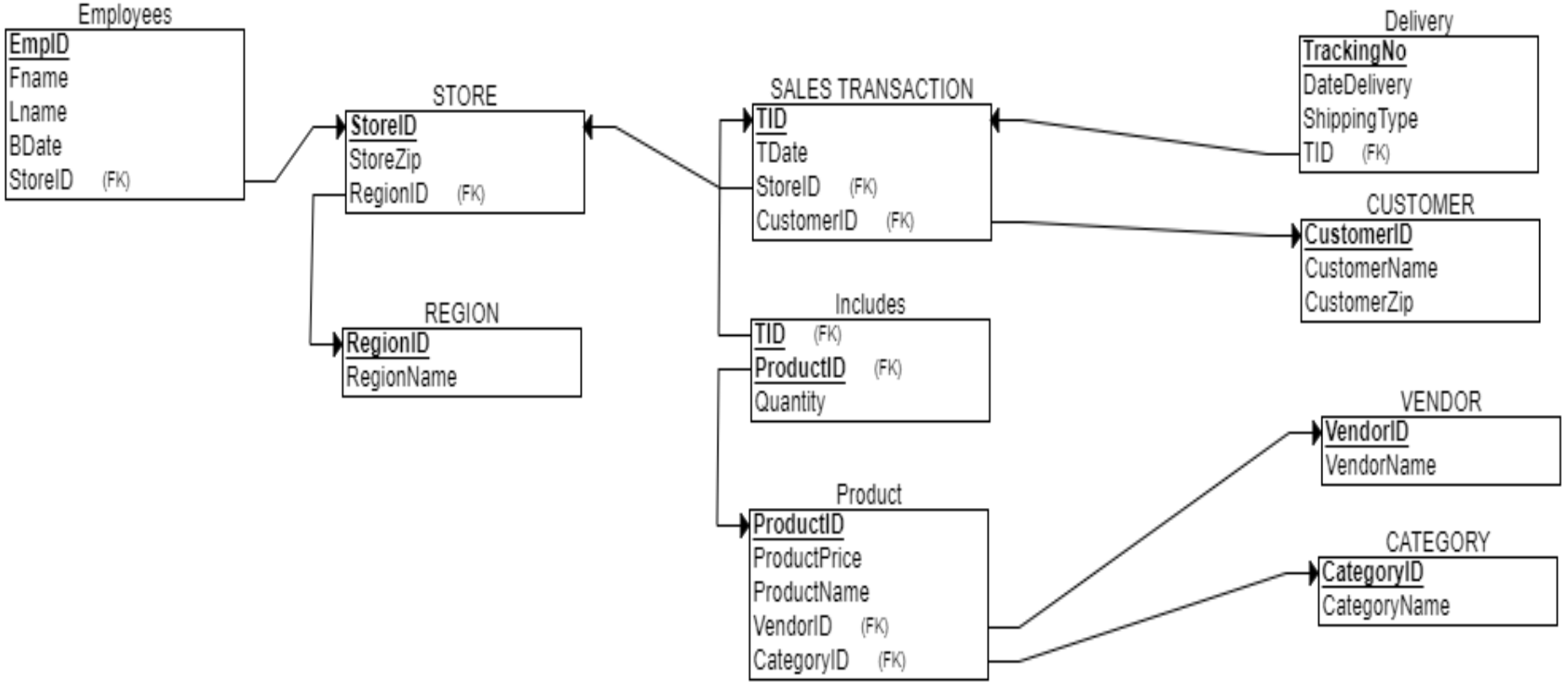
customerid	customername	customerzip
1-2-333	Tina	60137
2-3-444	Tony	60611
3-4-555	Pam	35401
NULL	NULL	NULL

7. TWO NEW ENTITIES

- For each employee who works at the store: an employeeID(unique), empname(First name & Last name), and Bdate.
- For each delivery: TrackingNo(unique), DateOfDelivery, and ShippingType.
- Each employee has to work in one store. Each store has one or more employees.
- Each sales transaction belongs to one delivery TrackingNo. One delivery TrackingNo can have one or more sales transactions.



8. NEW RELATIONAL SCHEMA



9. NEW RELATIONAL DATABASE CREATION – SQL

```
CREATE TABLE vendor(
 vendorid CHAR(2) NOT NULL,
 vendorname VARCHAR(25) NOT NULL,
 PRIMARY KEY(vendorid)
 );

CREATE TABLE category(
 categoryid CHAR(2) NOT NULL,
 categoryname VARCHAR(25) NOT NULL,
 PRIMARY KEY(categoryid)
 );

CREATE TABLE product(
 productid CHAR(3) NOT NULL,
 productname VARCHAR(25) NOT NULL,
 productprice NUMERIC(7, 2) NOT NULL,
 vendorid CHAR(2) NOT NULL,
 categoryid CHAR(2) NOT NULL,
 PRIMARY KEY(productid),
 FOREIGN KEY(vendorid) REFERENCES vendor(vendorid),
 FOREIGN KEY(categoryid) REFERENCES category(categoryid)
 );

CREATE TABLE region(
 regionid CHAR(1) NOT NULL,
 regionname VARCHAR(25) NOT NULL,
 PRIMARY KEY(regionid)
 );

CREATE TABLE store(
 storeid VARCHAR(3) NOT NULL,
 storezip CHAR(5) NOT NULL,
 regionid CHAR(1) NOT NULL,
 PRIMARY KEY(storeid),
 FOREIGN KEY(regionid) REFERENCES region(regionid)
 );

CREATE TABLE customer(
 customerid CHAR(7) NOT NULL,
 customername VARCHAR(15) NOT NULL,
 customerzip CHAR(5) NOT NULL,
 PRIMARY KEY(customerid)
 );

CREATE TABLE salestransaction(
 tid VARCHAR(8) NOT NULL,
 customerid CHAR(7) NOT NULL,
 storeid VARCHAR(3) NOT NULL,
 tdate DATE NOT NULL,
 PRIMARY KEY(tid),
 FOREIGN KEY(customerid) REFERENCES customer(customerid),
 FOREIGN KEY(storeid) REFERENCES store(storeid)
 );

CREATE TABLE includes(
 productid CHAR(3) NOT NULL,
 tid VARCHAR(8) NOT NULL,
 quantity INT NOT NULL,
 PRIMARY KEY(productid, tid),
 FOREIGN KEY(productid) REFERENCES product(productid),
 FOREIGN KEY(tid) REFERENCES salestransaction(tid)
 );

CREATE TABLE Delivery(
 trackingno VARCHAR(8) NOT NULL,
 datedelivery DATE NOT NULL,
 shippingtype VARCHAR(15) NOT NULL,
 tid VARCHAR(8) NOT NULL,
 PRIMARY KEY(trackingno),
 FOREIGN KEY(tid) REFERENCES salestransaction(tid)
 );

CREATE TABLE Employees(
 empid VARCHAR(5) NOT NULL,
 fname VARCHAR(25) NOT NULL,
 lname VARCHAR(25) NOT NULL,
 bdate DATE NOT NULL,
 storeid VARCHAR(3) NOT NULL,
 PRIMARY KEY(empid),
 FOREIGN KEY(storeid) REFERENCES store(storeid)
 );
```

10. POPULATING THE DATABASE WITH OUR OWN DATA

```
INSERT INTO vendor VALUES ('SO','Sony');
INSERT INTO vendor VALUES ('MS','Microsoft');
INSERT INTO category VALUES ('VG','Video Games');
INSERT INTO category VALUES ('EC','Entertainment & Computers');
INSERT INTO category VALUES ('SW','Software');
INSERT INTO product VALUES ('1X1','Playstation 5',500,'SO','VG');
INSERT INTO product VALUES ('2X2','Microsoft office',70,'MS','SW');
INSERT INTO product VALUES ('3X3','50 inch 4k TV',450,'SO','EC');
INSERT INTO product VALUES ('4X4','Xbox Series S',300,'MS','VG');
INSERT INTO product VALUES ('5X5','Microsoft Surface Tablet',250,'MS','EC');
INSERT INTO region VALUES ('B','Bay Area');
INSERT INTO region VALUES ('S','Sacramento Valley');
INSERT INTO store VALUES ('S1','94720','B');
INSERT INTO store VALUES ('S2','94016','B');
INSERT INTO store VALUES ('S3','95819','S');
INSERT INTO customer VALUES ('1-2-333','Bob','95678');
INSERT INTO customer VALUES ('2-3-444','Jack','94105');
INSERT INTO customer VALUES ('3-4-555','Emily','94204');
INSERT INTO salestransaction VALUES ('T111','1-2-333','S1','2020-01-04');
INSERT INTO salestransaction VALUES ('T222','2-3-444','S2','2021-01-28');
INSERT INTO salestransaction VALUES ('T333','1-2-333','S3','2021-11-02');
INSERT INTO salestransaction VALUES ('T444','3-4-555','S3','2022-02-14');
INSERT INTO salestransaction VALUES ('T555','2-3-444','S3','2022-03-05');
INSERT INTO includes VALUES ('1X1','T111',1);
INSERT INTO includes VALUES ('2X2','T222',1);
INSERT INTO includes VALUES ('3X3','T333',5);
INSERT INTO includes VALUES ('1X1','T333',1);
INSERT INTO includes VALUES ('4X4','T444',1);
INSERT INTO includes VALUES ('2X2','T444',2);
INSERT INTO includes VALUES ('4X4','T555',4);
INSERT INTO includes VALUES ('5X5','T555',2);
INSERT INTO Delivery VALUES ('12345678','2020-01-06','Expedited','T111');
INSERT INTO Delivery VALUES ('11111111','2021-01-29','Overnight','T222');
INSERT INTO Delivery VALUES ('87654321','2021-11-02','Local Pickup','T333');
INSERT INTO Delivery VALUES ('1234','2022-02-18','Expedited','T444');
INSERT INTO Delivery VALUES ('123456','2022-03-06','Overnight','T555');
INSERT INTO Employees VALUES ('11111','John','Doe','1996-03-23','S1');
INSERT INTO Employees VALUES ('22222','Jane','Doe','2000-10-07','S1');
INSERT INTO Employees VALUES ('33333','Robert','Bob','2001-09-15','S2');
INSERT INTO Employees VALUES ('44444','Summer','Johnson','1989-05-06','S2');
INSERT INTO Employees VALUES ('55555','Julia','Smith','1992-04-06','S3');
INSERT INTO Employees VALUES ('66666','Randall','Bowers','1985-07-30','S3');
```

11. TWO SQL QUERIES WITH OUR NEWLY ADDED DATA

Finds the average shipping time in days of each shippingtype and orders them from quickest to slowest.:

```
SELECT d.shippingtype, AVG(DATEDIFF(d.datedelivery, s.tdate))
FROM Delivery d, salestransaction s
WHERE s.tid = d.tid
GROUP BY shippingtype
ORDER BY AVG(DATEDIFF(d.datedelivery, s.tdate));
```

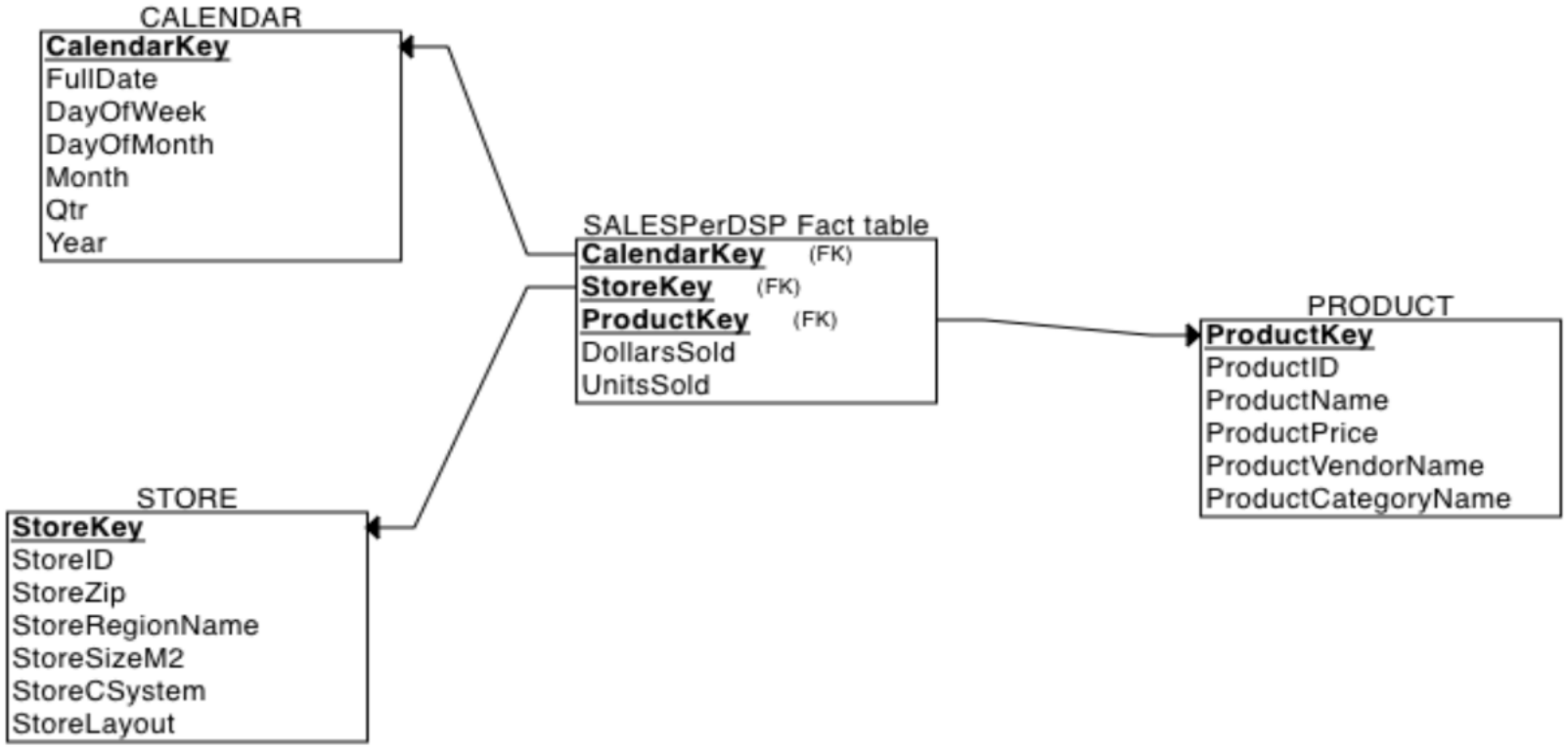
shippingtype	AVG(DATEDIFF(d.datedelivery, s.tdate))
Local Pickup	0.0000
Overnight	1.0000
Expedited	3.0000

Finds the total revenue of each region and sorts by revenue:

```
SELECT r.regionname, SUM(p.productprice * i.quantity)
FROM product p, salestransaction st, store s, includes i, region r
WHERE st.tid = i.tid AND i.productid = p.productid AND s.storeid = st.storeid AND s.regionid = r.regionid
GROUP BY regionname
ORDER BY SUM(p.productprice * i.quantity);
```

regionname	SUM(p.productprice * i.quantity)
Bay Area	570.00
Sacramento Valley	4890.00

12. (E8.1B) ANALYTICAL DATABASE SCHEMA



12. (E8.1B) ANALYTICAL DATABASE CREATION – SQL

```
CREATE TABLE CALENDAR(
 CalendarKey INT NOT NULL,
 FullDate DATE NOT NULL,
 DayOfWeek CHAR(15) NOT NULL,
 DayOfMonth INT NOT NULL,
 Month CHAR(10) NOT NULL,
 Qtr CHAR(2) NOT NULL,
 Year INT NOT NULL,
 PRIMARY KEY(CalendarKey)
 );

CREATE TABLE STORE(
 StoreKey INT NOT NULL,
 StoreID CHAR(5) NOT NULL,
 StoreZip CHAR(5) NOT NULL,
 StoreRegionName CHAR(15) NOT NULL,
 StoreSizeM2 INT NOT NULL,
 StoreCSystem CHAR(15) NOT NULL,
 StoreLayout CHAR(15) NOT NULL,
 PRIMARY KEY(StoreKey)
 );

CREATE TABLE PRODUCT(
 ProductKey INT NOT NULL,
 ProductID CHAR(5) NOT NULL,
 ProductName CHAR(25) NOT NULL,
 ProductPrice NUMERIC(7, 2) NOT NULL,
 ProductVendorName CHAR(25) NOT NULL,
 ProductCategoryName CHAR(25) NOT NULL,
 PRIMARY KEY(ProductKey)
 );
```

```
CREATE TABLE SALESPerDSP_Fact_table(
 CalendarKey INT NOT NULL,
 StoreKey INT NOT NULL,
 ProductKey INT NOT NULL,
 DollarsSold NUMERIC(10, 2) NOT NULL,
 UnitsSold INT NOT NULL,
 PRIMARY KEY(CalendarKey, StoreKey, ProductKey),
 FOREIGN KEY(CalendarKey) REFERENCES
CALENDAR(CalendarKey),
 FOREIGN KEY(StoreKey) REFERENCES STORE(StoreKey),
 FOREIGN KEY(ProductKey) REFERENCES PRODUCT(ProductKey)
 );
```

12. (E8.1C) POPULATING THE ANALYTICAL DATABASE - SQL

```
INSERT INTO CALENDAR VALUES (1, '2020-01-01', 'Wednesday', '1', 'January', '01', '2020');
INSERT INTO CALENDAR VALUES (2, '2020-01-02', 'Thursday', '2', 'January', '01', '2020');
INSERT INTO CALENDAR VALUES (3, '2020-01-03', 'Friday', '3', 'January', '01', '2020');
INSERT INTO STORE VALUES (1, 'S1', '60600', 'Chicagoland', 51000, 'Cashiers', 'Modern');
INSERT INTO STORE VALUES (2, 'S2', '60605', 'Chicagoland', 35000, 'Self Service', 'Traditional');
INSERT INTO STORE VALUES (3, 'S3', '35400', 'Tristate', 55000, 'Mixed', 'Traditional');
INSERT INTO PRODUCT VALUES (1, '1X1', 'Zzz Bag', 100, 'Pacifica Gear', 'Camping');
INSERT INTO PRODUCT VALUES (2, '2X2', 'Easy Boot', 70, 'Mountain King', 'Footwear');
INSERT INTO PRODUCT VALUES (3, '3X3', 'Cosy Sock', 15, 'Mountain King', 'Footwear');
INSERT INTO PRODUCT VALUES (4, '4X4', 'Dura Boot', 90, 'Pacifica Gear', 'Footwear');
INSERT INTO PRODUCT VALUES (5, '5X5', 'Tiny Tent', 150, 'Mountain King', 'Camping');
INSERT INTO PRODUCT VALUES (6, '6X6', 'Biggy Tent', 250, 'Mountain King', 'Camping');
INSERT INTO SALESPerDSP_Fact_table VALUES (1, 3, 6, 250, 1);
INSERT INTO SALESPerDSP_Fact_table VALUES (1, 3, 5, 300, 2);
INSERT INTO SALESPerDSP_Fact_table VALUES (1, 3, 1, 100, 1);
INSERT INTO SALESPerDSP_Fact_table VALUES (2, 1, 2, 70, 1);
INSERT INTO SALESPerDSP_Fact_table VALUES (2, 2, 3, 45, 3);
INSERT INTO SALESPerDSP_Fact_table VALUES (2, 2, 4, 180, 2);
INSERT INTO SALESPerDSP_Fact_table VALUES (3, 2, 3, 120, 8);
INSERT INTO SALESPerDSP_Fact_table VALUES (3, 2, 4, 90, 1);
INSERT INTO SALESPerDSP_Fact_table VALUES (3, 3, 3, 60, 4);
```

13. ANALYTICAL SQL QUERIES

How many units did each region sell?

```
SELECT StoreRegionName, SUM(UnitsSold)
FROM STORE
LEFT JOIN SALESPerDSP_Fact_table on SALESPerDSP_Fact_table.StoreKey = STORE.StoreKey
GROUP BY StoreRegionName;
```

OUTPUT

StoreRegionName	SUM(UnitsSold)
Chicagoland	15
Tristate	8

On Jan 02 2020 how many units of each product were sold?

```
SELECT FullDate, ProductName, UnitsSold
FROM PRODUCT
LEFT JOIN SALESPerDSP_Fact_table ON SALESPerDSP_Fact_table.ProductKey = PRODUCT.ProductKey
LEFT JOIN CALENDAR on CALENDAR.CalendarKey = SALESPerDSP_Fact_table.CalendarKey
WHERE FullDate LIKE '2020-01-02';
```

OUTPUT

FullDate	ProductName	UnitsSold
2020-01-02	Easy Boot	1
2020-01-02	Cosy Sock	3
2020-01-02	Dura Boot	2

13. ANALYTICAL SQL QUERIES

On days when Cosy Socks sold, how many units sold and what were the combined DollarsSold?

```
SELECT ProductName, FullDate, SUM(UnitsSold), SUM(DollarsSold)
FROM CALENDAR
LEFT JOIN SALESPerDSP_Fact_table ON SALESPerDSP_Fact_table.CalendarKey = CALENDAR.CalendarKey
LEFT JOIN PRODUCT on PRODUCT.ProductKey = SALESPerDSP_Fact_table.ProductKey
WHERE ProductName LIKE "Cosy Sock"
GROUP BY ProductName, FullDate;
```

OUPUT

ProductName	FullDate	SUM(UnitsSold)	SUM(DollarsSold)
Cosy Sock	2020-01-02	3	45
Cosy Sock	2020-01-03	12	180

What day & store had the most dollars sold of what product and what was the amount?

```
SELECT StoreID, FullDate, ProductName, DollarsSold
FROM STORE
LEFT JOIN SALESPerDSP_Fact_table ON SALESPerDSP_Fact_table.StoreKey = STORE.StoreKey
LEFT JOIN CALENDAR on CALENDAR.CalendarKey = SALESPerDSP_Fact_table.CalendarKey
LEFT JOIN PRODUCT on PRODUCT.ProductKey = SALESPerDSP_Fact_table.ProductKey
GROUP BY StoreID, FullDate, ProductName, DollarsSold
ORDER BY DollarsSold DESC
LIMIT 1;
```

OUTPUT

StoreID	FullDate	ProductName	DollarsSold
S3	2020-01-01	Tiny Tent	300

13. ANALYTICAL SQL QUERIES

What were the highest selling products, how many units were sold and what region?

```
SELECT StoreRegionName, PRODUCT.ProductKey, PRODUCT.ProductName, SUM(UnitsSold)
FROM STORE
LEFT JOIN SALESPerDSP_Fact_table on SALESPerDSP_Fact_table.StoreKey = STORE.StoreKey
LEFT JOIN PRODUCT on PRODUCT.ProductKey = SALESPerDSP_Fact_table.ProductKey
GROUP BY StoreRegionName, ProductKey
ORDER BY SUM(UnitsSold) DESC;
```

OUTPUT

StoreRegionName	ProductKey	ProductName	SUM(UnitsSold)
Chicagoland	3	Cosy Sock	11
Tristate	3	Cosy Sock	4
Chicagoland	4	Dura Boot	3
Tristate	5	Tiny Tent	2
Chicagoland	2	Easy Boot	1
Tristate	1	Zzz Bag	1
Tristate	6	Biggy Tent	1