Siddhartha Shankar, MS-BA
NET ID: SXS180085

Project Report - Database Foundations for Business Analytics

BUAN 6320.002

Normalizing the data

The provided three datasets are downloaded, which are provided in .txt format. Data is loaded in R and data frames are created by normalizing the data using the Data Dictionary. The details of the table fields are as follows: -

Table 1: Brand (**BRAND_ID**, BRAND_NAME, BRAND_TYPE)

Table 2: Customer (**CUST_CODE**, CUST_FNAME, CUST_LNAME, CUST_STREET, CUST_CITY, CUST_STATE, CUST_ZIP, CUST_BALANCE)

Table 3: Department (**DEPT_NUM**, DEPT_NAME, DEPT_MAILBOX, DEPT_PHONE)

[Note: - EMP_NUM is not included in the Department table because there is one to many relations between DEPT_NUM and EMP_NUM. DEPT_NUM is mentioned as foreign key in Employee table for reference.]

Table 4: Employee (**EMP_NUM**, EMP_FNAME, EMP_LNAME, EMP_EMAIL, EMP_PHONE, EMP_HIREDATE, EMP_TITLE, EMP_COMM, DEPT_NUM)

Table 5: Invoice (**INV_NUM**, INV_DATE, CUST_CODE, INV_TOTAL, EMPLOYEE_ID)

Table 6: Product (**PROD_SKU**, PROD_DESCRIPT, PROD_TYPE, PROD_BASE, PROD_CATEGORY, PROD_PRICE, PROD_QOH, PROD_MIN, BRAND_ID)

Table 7: Line (**INV_NUM**, **LINE_NUM**, PROD_SKU, LINE_QTY, LINE_PRICE)

[Note: - INV_NUM, LINE_NUM is declared as primary key as the combination of both is unique]

Table 8: Salary_History (**EMP_NUM**, **SAL_FROM**, SAL_END, SAL_AMOUNT)

Table 9: Vendor (**VEND_ID**, VEND_NAME, VEND_STREET, VEND_CITY, VEND_STATE, VEND_ZIP)

Table 10: Supplies (**PROD_SKU, VEND_ID**)

Cleaning the data

Table 1: Brand - Duplicate values are removed from the BRAND_ID column

Table 2: Customer – Duplicate values are removed from the CUST_CODE column

Table 3: Department – Duplicate values are removed from the DEPT_NUM column

Table 4: Employee

i. Duplicate values are removed from the EMP_NUM column
ii. Days are converted into date keeping origin date as “1899-12-31”in EMP_HIREDATE column.

Table 5: Invoice

i. Duplicate values are removed from the INV_NUM column.
ii. Days are converted into date keeping origin date as “1899-12-31” in INV_DATE column.

Table 6: Product

i. Duplicate values are removed from the PROD_SKU column.
ii. Special characters in PROD_QOH is removed using gsub function in R. As R return String data type with gsub function, it is converted into numeric data type using as.numeric function.
iii. Similarly, Special Characters in PROD_SKU is removed using gsub function.

Table 7: Line – Unique Values of data are taken i.e. duplicate rows are removed

Table 8: Salary_history

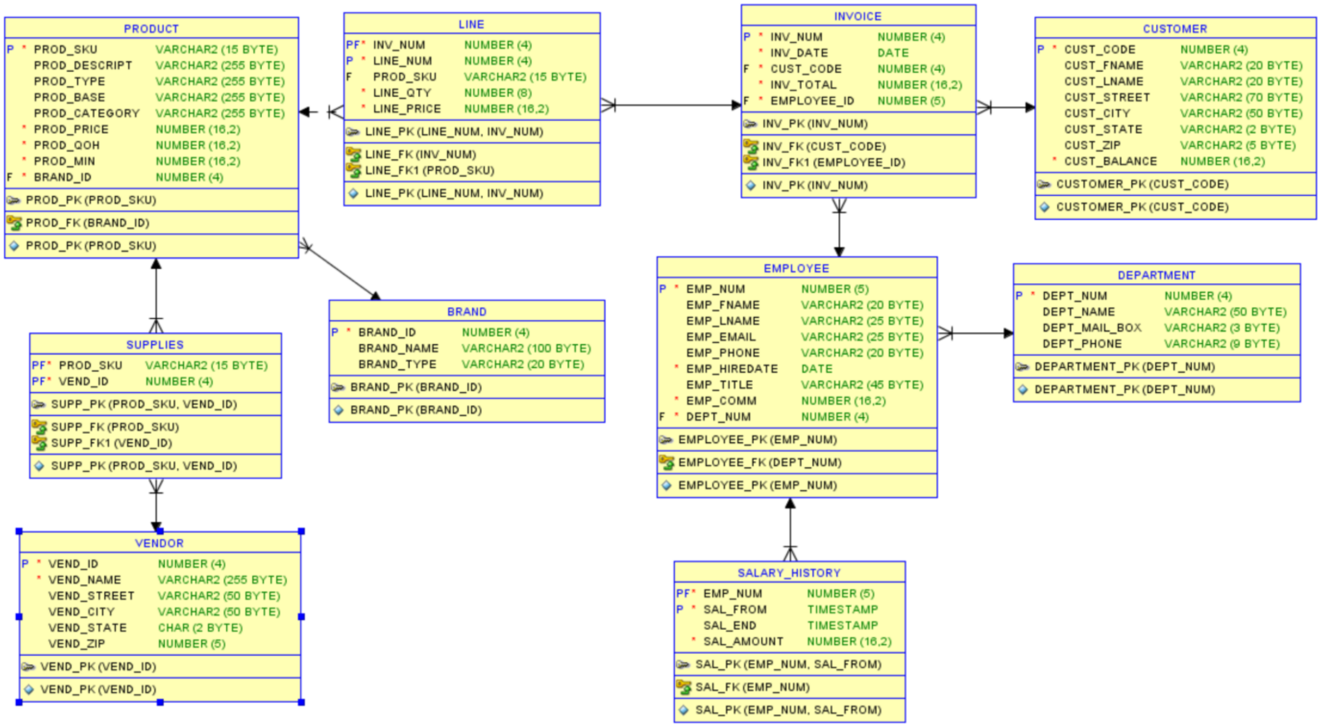
i. First, SAL_END is converted into characters with as.character function in R. Then , ‘-‘ character is removed.
ii. Dummy value ‘0’ is inserted in place of ‘-‘ and the data type is converted into integer for SAL_END column. Once the table Salary_History is inserted, using Update function SAL_END = ‘1899’ is removed.
iii. Days are converted into date keeping origin date as “1899-12-31” in SAL_FROM and SAL_END columns.

Table 9: Vendor - Duplicate values are removed from the VEND_ID column

Table 10: Supplies – Special characters in PROD_SKU is removed using gsub function in R.

Logical and physical model is created using Data Modeler in Oracle SQL Developer

Primary key, Secondary key is evident as per the below diagram:



```
PRODUCT
P  *  PROD_SKU          VARCHAR2 (15 BYTE)
      PROD_DESCRIPT     VARCHAR2 (255 BYTE)
      PROD_TYPE         VARCHAR2 (255 BYTE)
      PROD_BASE         VARCHAR2 (255 BYTE)
      PROD_CATEGORY     VARCHAR2 (255 BYTE)
   *  PROD_PRICE        NUMBER (16,2)
   *  PROD_QOH          NUMBER (16,2)
   *  PROD_MIN          NUMBER (16,2)
F  *  BRAND_ID          NUMBER (4)
PROD_PK (PROD_SKU)
PROD_FK (BRAND_ID)
PROD_PK (PROD_SKU)
```

```
LINE
PF*  INV_NUM       NUMBER (4)
P *  LINE_NUM      NUMBER (4)
F    PROD_SKU      VARCHAR2 (15 BYTE)
  *  LINE_QTY      NUMBER (8)
  *  LINE_PRICE    NUMBER (16,2)
LINE_PK (LINE_NUM, INV_NUM)
LINE_FK (INV_NUM)
LINE_FK1 (PROD_SKU)
LINE_PK (LINE_NUM, INV_NUM)
```

```
INVOICE
P *  INV_NUM        NUMBER (4)
  *  INV_DATE       DATE
F *  CUST_CODE      NUMBER (4)
  *  INV_TOTAL      NUMBER (16,2)
F *  EMPLOYEE_ID    NUMBER (5)
INV_PK (INV_NUM)
INV_FK (CUST_CODE)
INV_FK1 (EMPLOYEE_ID)
INV_PK (INV_NUM)
```

```
CUSTOMER
P *  CUST_CODE      NUMBER (4)
     CUST_FNAME     VARCHAR2 (20 BYTE)
     CUST_LNAME     VARCHAR2 (20 BYTE)
     CUST_STREET    VARCHAR2 (70 BYTE)
     CUST_CITY      VARCHAR2 (50 BYTE)
     CUST_STATE     VARCHAR2 (2 BYTE)
     CUST_ZIP       VARCHAR2 (5 BYTE)
  *  CUST_BALANCE   NUMBER (16,2)
CUSTOMER_PK (CUST_CODE)
CUSTOMER_PK (CUST_CODE)
```

```
SUPPLIES
PF*  PROD_SKU    VARCHAR2 (15 BYTE)
PF*  VEND_ID     NUMBER (4)
SUPP_PK (PROD_SKU, VEND_ID)
SUPP_FK (PROD_SKU)
SUPP_FK1 (VEND_ID)
SUPP_PK (PROD_SKU, VEND_ID)
```

```
BRAND
P *  BRAND_ID      NUMBER (4)
     BRAND_NAME    VARCHAR2 (100 BYTE)
     BRAND_TYPE    VARCHAR2 (20 BYTE)
BRAND_PK (BRAND_ID)
BRAND_PK (BRAND_ID)
```

```
EMPLOYEE
P *  EMP_NUM        NUMBER (5)
     EMP_FNAME      VARCHAR2 (20 BYTE)
     EMP_LNAME      VARCHAR2 (25 BYTE)
     EMP_EMAIL      VARCHAR2 (25 BYTE)
     EMP_PHONE      VARCHAR2 (20 BYTE)
  *  EMP_HIREDATE   DATE
     EMP_TITLE      VARCHAR2 (45 BYTE)
  *  EMP_COMM       NUMBER (16,2)
F *  DEPT_NUM       NUMBER (4)
EMPLOYEE_PK (EMP_NUM)
EMPLOYEE_FK (DEPT_NUM)
EMPLOYEE_PK (EMP_NUM)
```

```
DEPARTMENT
P *  DEPT_NUM        NUMBER (4)
     DEPT_NAME       VARCHAR2 (50 BYTE)
     DEPT_MAIL_BOX   VARCHAR2 (3 BYTE)
     DEPT_PHONE      VARCHAR2 (9 BYTE)
DEPARTMENT_PK (DEPT_NUM)
DEPARTMENT_PK (DEPT_NUM)
```

```
VENDOR
P *  VEND_ID        NUMBER (4)
  *  VEND_NAME      VARCHAR2 (255 BYTE)
     VEND_STREET    VARCHAR2 (50 BYTE)
     VEND_CITY      VARCHAR2 (50 BYTE)
     VEND_STATE     CHAR (2 BYTE)
     VEND_ZIP       NUMBER (5)
VEND_PK (VEND_ID)
VEND_PK (VEND_ID)
```

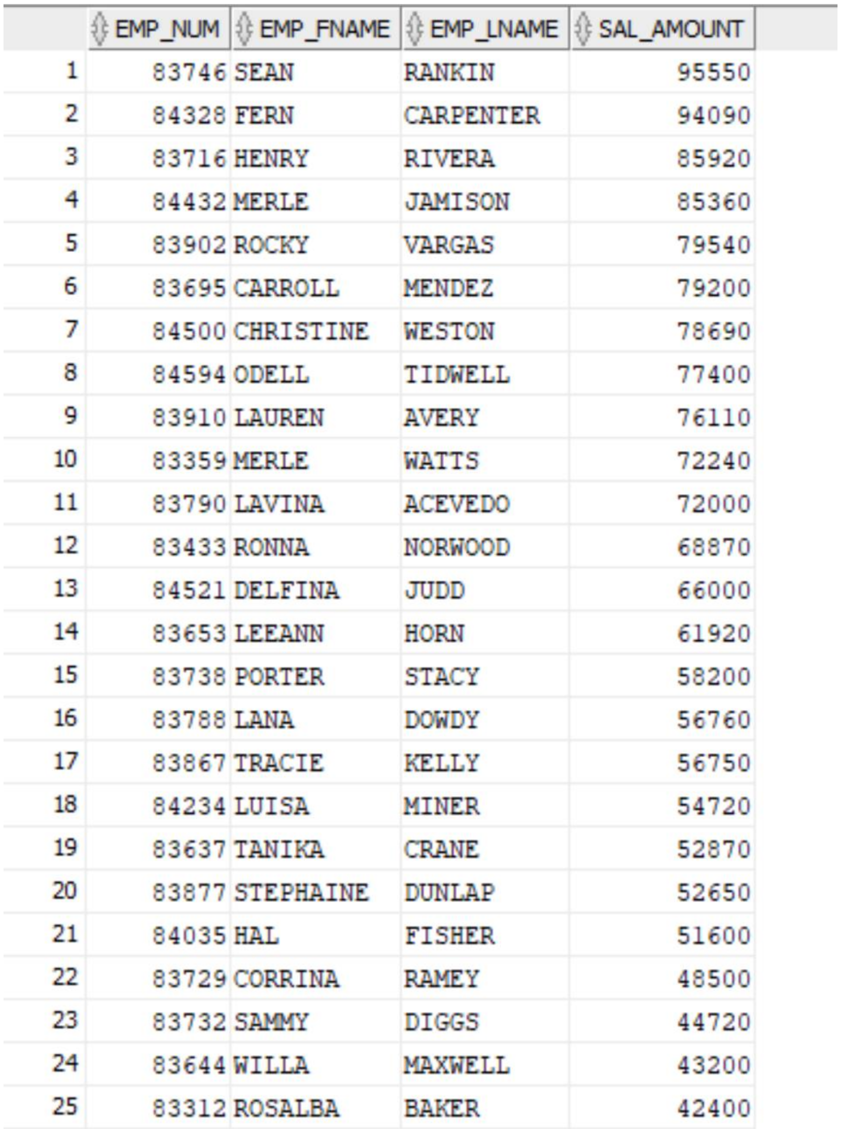
```
SALARY_HISTORY
PF*  EMP_NUM      NUMBER (5)
P *  SAL_FROM     TIMESTAMP
     SAL_END      TIMESTAMP
  *  SAL_AMOUNT   NUMBER (16,2)
SAL_PK (EMP_NUM, SAL_FROM)
SAL_FK (EMP_NUM)
SAL_PK (EMP_NUM, SAL_FROM)
```

The cleaned data is inserted in Oracle SQL developer with all required CONSTRAINTS, PRIMARY KEY AND FOREIGN KEY.

```
CREATE TABLE BRAND (

        BRAND_ID      NUMBER(4) NOT NULL,

        BRAND_NAME    VARCHAR2(100) ,

        BRAND_TYPE    VARCHAR(20),

        CONSTRAINT BRAND_PK PRIMARY KEY (BRAND_ID)

   );


CREATE TABLE CUSTOMER (
        CUST_CODE      NUMBER(4) NOT NULL,
        CUST_FNAME    VARCHAR2(20) ,
        CUST_LNAME    VARCHAR2(20),
        CUST_STREET   VARCHAR2(70),
        CUST_CITY     VARCHAR2(50),
        CUST_STATE    VARCHAR2(2),
        CUST_ZIP      VARCHAR2(5),
        CUST_BALANCE  DECIMAL(16,2) NOT NULL,
        CONSTRAINT CUSTOMER_PK PRIMARY KEY (CUST_CODE)
   );
CREATE TABLE DEPARTMENT (
        DEPT_NUM      NUMBER(4) NOT NULL,
        DEPT_NAME    VARCHAR(50) ,
        DEPT_MAIL_BOX    VARCHAR(3),
        DEPT_PHONE   VARCHAR(9),
        CONSTRAINT DEPARTMENT_PK PRIMARY KEY (DEPT_NUM)
   );

CREATE TABLE EMPLOYEE (
        EMP_NUM      NUMBER(5) NOT NULL,
        EMP_FNAME    VARCHAR(20) ,
        EMP_LNAME    VARCHAR(25),
        EMP_EMAIL    VARCHAR(25),
        EMP_PHONE    VARCHAR(20),
        EMP_HIREDATE  DATE NOT NULL,
        EMP_TITLE      VARCHAR(45),
        EMP_COMM  DECIMAL(16,2) NOT NULL,
        DEPT_NUM NUMBER(4) NOT NULL,

        CONSTRAINT EMPLOYEE_PK PRIMARY KEY (EMP_NUM),

        CONSTRAINT EMPLOYEE_FK FOREIGN KEY (DEPT_NUM) REFERENCES DEPARTMENT (DEPT_NUM)

  );


CREATE TABLE INVOICE(

        INV_NUM  NUMBER(4) NOT NULL,

        INV_DATE DATE NOT NULL,

        CUST_CODE NUMBER(4) NOT NULL,

        INV_TOTAL DECIMAL(16,2) NOT NULL,

        EMPLOYEE_ID NUMBER(5) NOT NULL,

        CONSTRAINT INV_PK PRIMARY KEY(INV_NUM),

        CONSTRAINT INV_FK FOREIGN KEY (CUST_CODE) REFERENCES CUSTOMER(CUST_CODE),

        CONSTRAINT INV_FK1 FOREIGN KEY (EMPLOYEE_ID) REFERENCES EMPLOYEE(EMP_NUM)

);


CREATE TABLE PRODUCT(

        PROD_SKU VARCHAR2(15) NOT NULL,

        PROD_DESCRIPT VARCHAR2(255) NULL,

        PROD_TYPE VARCHAR2(255) NULL,

        PROD_BASE VARCHAR2(255) NULL,

        PROD_CATEGORY VARCHAR2(255) NULL,

        PROD_PRICE DECIMAL(16,2) NOT NULL,

        PROD_QOH  DECIMAL(16,2) NOT NULL,

        PROD_MIN  DECIMAL(16,2) NOT NULL,

        BRAND_ID NUMBER(4) NOT NULL,
        CONSTRAINT PROD_PK PRIMARY KEY(PROD_SKU),
        CONSTRAINT PROD_FK FOREIGN KEY (BRAND_ID) REFERENCES BRAND(BRAND_ID)
);

CREATE TABLE LINE(
        INV_NUM NUMBER(4) NOT NULL,
        LINE_NUM NUMBER(4) NOT NULL,
        PROD_SKU VARCHAR2(15) NULL,
        LINE_QTY NUMBER(8) NOT NULL,
        LINE_PRICE DECIMAL(16,2) NOT NULL,
        CONSTRAINT LINE_PK PRIMARY KEY(LINE_NUM,INV_NUM),
        CONSTRAINT LINE_FK FOREIGN KEY (INV_NUM) REFERENCES INVOICE(INV_NUM),
        CONSTRAINT LINE_FK1 FOREIGN KEY (PROD_SKU) REFERENCES PRODUCT(PROD_SKU)
);

CREATE TABLE SALARY_HISTORY(
        EMP_NUM NUMBER(5) NOT NULL,
        SAL_FROM TIMESTAMP NOT NULL,
        SAL_END TIMESTAMP NULL,
        SAL_AMOUNT DECIMAL(16,2) NOT NULL,
        CONSTRAINT SAL_PK PRIMARY KEY(EMP_NUM,SAL_FROM),
        CONSTRAINT SAL_FK FOREIGN KEY (EMP_NUM) REFERENCES EMPLOYEE(EMP_NUM)
);

CREATE TABLE VENDOR(
        VEND_ID NUMBER(4) NOT NULL,
        VEND_NAME VARCHAR2(255) NOT NULL,
        VEND_STREET VARCHAR2(50) NULL,
        VEND_CITY VARCHAR2(50) NULL,
        VEND_STATE CHAR(2) NULL,
        VEND_ZIP NUMBER(5) NULL,
        CONSTRAINT VEND_PK PRIMARY KEY(VEND_ID)
);

CREATE TABLE SUPPLIES(
        PROD_SKU VARCHAR2(15) NULL,
        VEND_ID NUMBER(4) NOT NULL,
        CONSTRAINT SUPP_PK PRIMARY KEY(PROD_SKU,VEND_ID),
        CONSTRAINT SUPP_FK FOREIGN KEY (PROD_SKU) REFERENCES PRODUCT(PROD_SKU),
        CONSTRAINT SUPP_FK1 FOREIGN KEY (VEND_ID) REFERENCES VENDOR(VEND_ID)
);
```

1. Write a query to display the current salary for each employee in department 300. Assume that only current employees are kept in the system, and therefore the most current salary for each employee is the entry in the salary history with a NULL end date. Sort the output in descending order by salary amount.

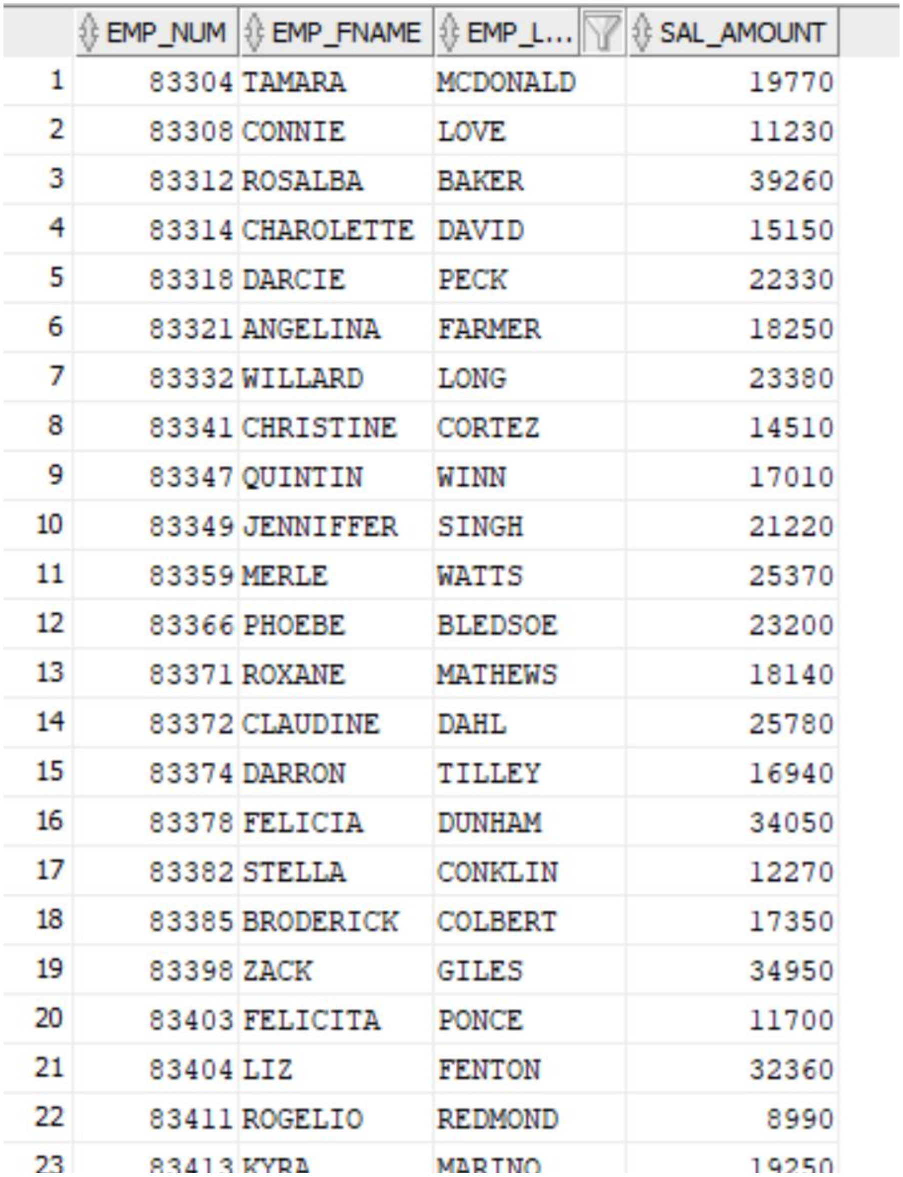
```
SELECT e.EMP_NUM, e.EMP_FNAME, e.EMP_LNAME, s.SAL_AMOUNT
FROM EMPLOYEE e
JOIN SALARY_HISTORY s ON s.EMP_NUM = e.EMP_NUM
WHERE e.DEPT_NUM = 300 AND s.SAL_END IS NULL
ORDER BY 4 DESC;
```



	EMP_NUM	EMP_FNAME	EMP_LNAME	SAL_AMOUNT
1	83746	SEAN	RANKIN	95550
2	84328	FERN	CARPENTER	94090
3	83716	HENRY	RIVERA	85920
4	84432	MERLE	JAMISON	85360
5	83902	ROCKY	VARGAS	79540
6	83695	CARROLL	MENDEZ	79200
7	84500	CHRISTINE	WESTON	78690
8	84594	ODELL	TIDWELL	77400
9	83910	LAUREN	AVERY	76110
10	83359	MERLE	WATTS	72240
11	83790	LAVINA	ACEVEDO	72000
12	83433	RONNA	NORWOOD	68870
13	84521	DELFINA	JUDD	66000
14	83653	LEEANN	HORN	61920
15	83738	PORTER	STACY	58200
16	83788	LANA	DOWDY	56760
17	83867	TRACIE	KELLY	56750
18	84234	LUISA	MINER	54720
19	83637	TANIKA	CRANE	52870
20	83877	STEPHAINE	DUNLAP	52650
21	84035	HAL	FISHER	51600
22	83729	CORRINA	RAMEY	48500
23	83732	SAMMY	DIGGS	44720
24	83644	WILLA	MAXWELL	43200
25	83312	ROSALBA	BAKER	42400

2. Write a query to display the starting salary for each employee. The starting salary would be the entry in the salary history with the oldest salary start date for each employee. Sort the output by employee number.

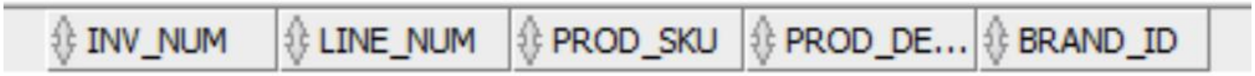
SELECT e.EMP_NUM,e.EMP_FNAME,e.EMP_LNAME,s.SAL_AMOUNT
FROM EMPLOYEE e
JOIN SALARY_HISTORY s ON s.EMP_NUM = e.EMP_NUM
WHERE s.SAL_FROM = e.EMP_HIREDATE
ORDER BY 1 ;



	EMP_NUM	EMP_FNAME	EMP_L...	SAL_AMOUNT
1	83304	TAMARA	MCDONALD	19770
2	83308	CONNIE	LOVE	11230
3	83312	ROSALBA	BAKER	39260
4	83314	CHAROLETTE	DAVID	15150
5	83318	DARCIE	PECK	22330
6	83321	ANGELINA	FARMER	18250
7	83332	WILLARD	LONG	23380
8	83341	CHRISTINE	CORTEZ	14510
9	83347	QUINTIN	WINN	17010
10	83349	JENNIFFER	SINGH	21220
11	83359	MERLE	WATTS	25370
12	83366	PHOEBE	BLEDSOE	23200
13	83371	ROXANE	MATHEWS	18140
14	83372	CLAUDINE	DAHL	25780
15	83374	DARRON	TILLEY	16940
16	83378	FELICIA	DUNHAM	34050
17	83382	STELLA	CONKLIN	12270
18	83385	BRODERICK	COLBERT	17350
19	83398	ZACK	GILES	34950
20	83403	FELICITA	PONCE	11700
21	83404	LIZ	FENTON	32360
22	83411	ROGELIO	REDMOND	8990
23	83413	KYRA	MARINO	19250

3. Write a query to display the invoice number, line numbers, product SKUs, product descriptions, and brand ID for sales of sealer and top coat products of the same brand on the same invoice.

```
SELECT
SEALER.INV_NUM,SEALER.LINE_NUM,SEALER.PROD_SKU,SEALER.PROD_DESCRIPT,SEALER.BRAND_ID

FROM

(SELECT l1.INV_NUM,l1.LINE_NUM,l1.PROD_SKU,p1.PROD_DESCRIPT,p1.BRAND_ID

FROM LINE l1,PRODUCT p1

WHERE l1.PROD_SKU = p1.PROD_SKU

AND p1.PROD_CATEGORY = 'SEALER'

) SEALER,

(SELECT l2.INV_NUM,l2.LINE_NUM,l2.PROD_SKU,p2.PROD_DESCRIPT,p2.BRAND_ID

FROM LINE l2,PRODUCT p2

WHERE l2.PROD_SKU = p2.PROD_SKU

AND p2.PROD_CATEGORY = 'TOP COAT'

) TOP_COAT

WHERE SEALER.INV_NUM = TOP_COAT.INV_NUM

AND SEALER.BRAND_ID = TOP_COAT.BRAND_ID

;
```



INV_NUM	LINE_NUM	PROD_SKU	PROD_DE...	BRAND_ID

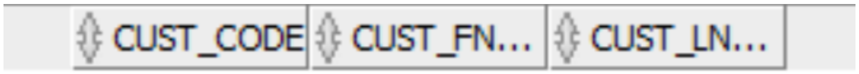
[NOTE : - Although, it is assumed that Invoice No. is Unique while cleansing the data]

4. The Binder Prime Company wants to recognize the employee who sold the most of their products during a specified period. Write a query to display the employee number, employee first name, employee last name, e-mail address, and total units sold for the employee who sold the most Binder Prime brand products between November 1, 2015, and December 5, 2015. If there is a tie for most units sold, sort the output by employee last name.

5. Write a query to display the customer code, first name, and last name of all customers who have had at least one invoice completed by employee 83649 and at least one invoice completed by employee 83677. Sort the output by customer last name and then first name.

```
SELECT * FROM
(SELECT c.CUST_CODE, c.CUST_FNAME, c.CUST_LNAME
FROM CUSTOMER c,INVOICE i
WHERE i.EMPLOYEE_ID = 83649
AND c.CUST_CODE = i.CUST_CODE)
INTERSECT
(SELECT c1.CUST_CODE, c1.CUST_FNAME, c1.CUST_LNAME FROM CUSTOMER c1,invoice
i1
WHERE i1.EMPLOYEE_ID = 83677
AND c1.CUST_CODE = i1.CUST_CODE)
ORDER BY 3,2;
```

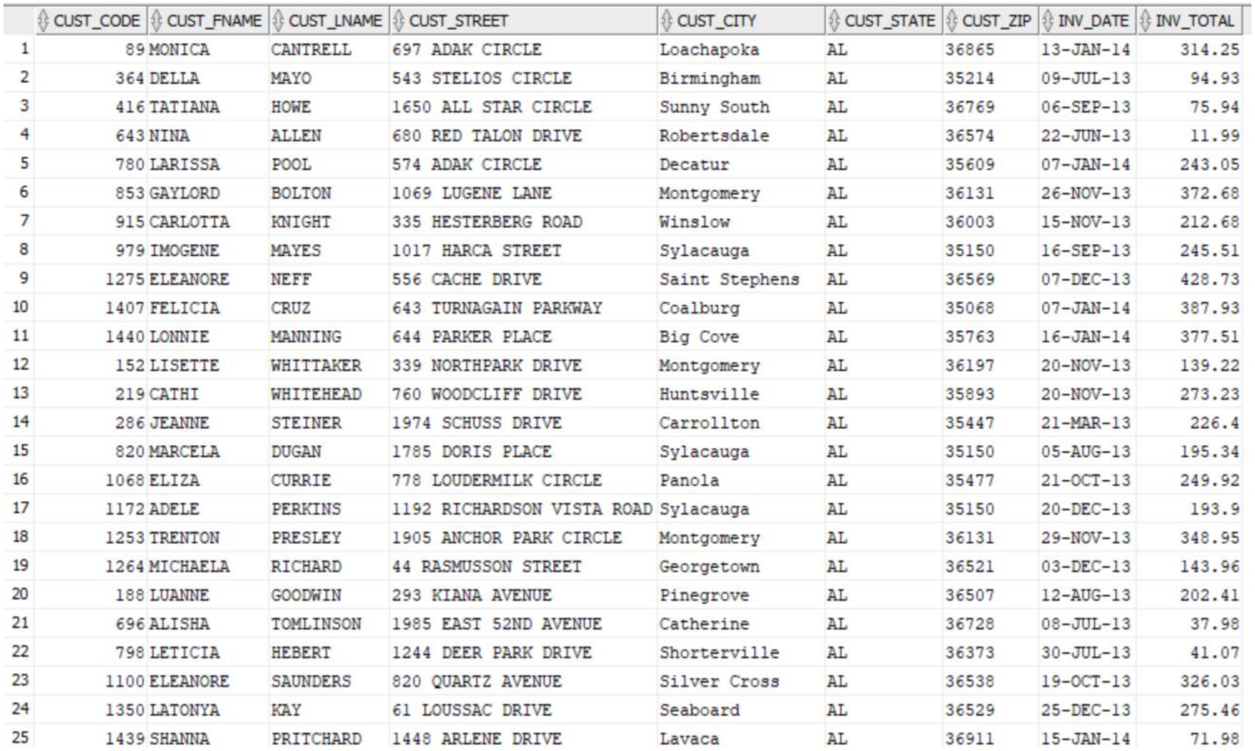
No results



CUST_CODE	CUST_FN...	CUST_LN...

6. LargeCo is planning a new promotion in Alabama (AL) and wants to know about the largest purchases made by customers in that state. Write a query to display the customer code, customer first name, last name, full address, invoice date, and invoice total of the largest purchase made by each customer in Alabama. Be certain to include any customers in Alabama who have never made a purchase (their invoice dates should be NULL and the invoice totals should display as 0).

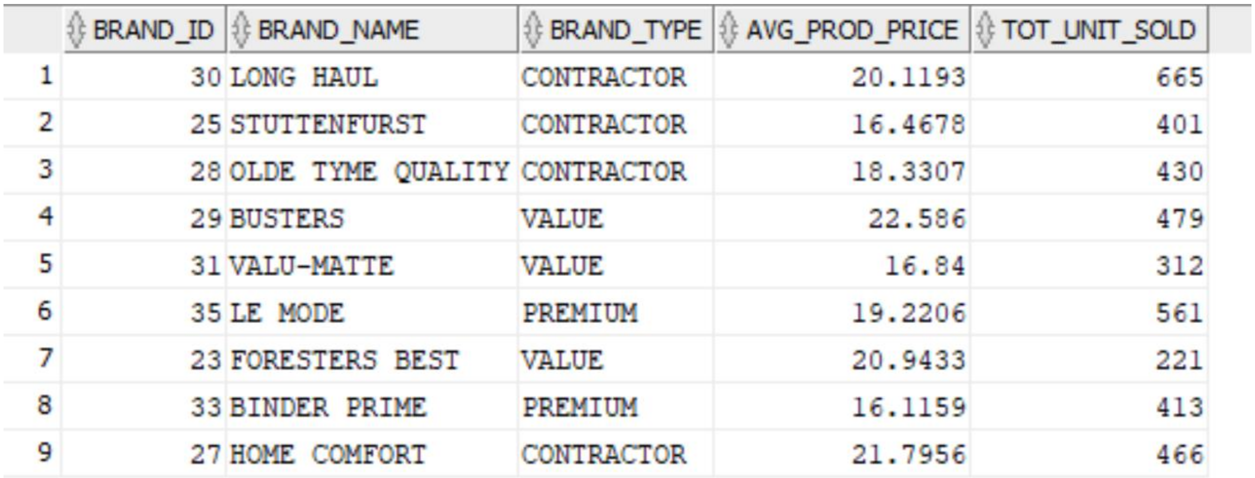
```
SELECT * FROM
(SELECT c.CUST_CODE, c.CUST_FNAME, c.CUST_LNAME,c.CUST_STREET,
c.CUST_CITY,c.CUST_STATE,c.CUST_ZIP,i.INV_DATE,COALESCE(i.INV_TOTAL,0)
AS INV_TOTAL
FROM  CUSTOMER C LEFT JOIN INVOICE i ON c.CUST_CODE = i.CUST_CODE
WHERE c.CUST_STATE = 'AL') A
WHERE (A.CUST_CODE,A.INV_TOTAL) IN
(
SELECT  c.CUST_CODE,NVL(MAX(I.INV_TOTAL),0)
FROM CUSTOMER c LEFT JOIN INVOICE i ON c.CUST_CODE = i.CUST_CODE
WHERE c.CUST_STATE = 'AL'
GROUP BY c.CUST_CODE
);
```



	CUST_CODE	CUST_FNAME	CUST_LNAME	CUST_STREET	CUST_CITY	CUST_STATE	CUST_ZIP	INV_DATE	INV_TOTAL
1	89	MONICA	CANTRELL	697 ADAK CIRCLE	Loachapoka	AL	36865	13-JAN-14	314.25
2	364	DELLA	MAYO	543 STELIOS CIRCLE	Birmingham	AL	35214	09-JUL-13	94.93
3	416	TATIANA	HOWE	1650 ALL STAR CIRCLE	Sunny South	AL	36769	06-SEP-13	75.94
4	643	NINA	ALLEN	680 RED TALON DRIVE	Robertsdale	AL	36574	22-JUN-13	11.99
5	780	LARISSA	POOL	574 ADAK CIRCLE	Decatur	AL	35609	07-JAN-14	243.05
6	853	GAYLORD	BOLTON	1069 LUGENE LANE	Montgomery	AL	36131	26-NOV-13	372.68
7	915	CARLOTTA	KNIGHT	335 HESTERBERG ROAD	Winslow	AL	36003	15-NOV-13	212.68
8	979	IMOGENE	MAYES	1017 HARCA STREET	Sylacauga	AL	35150	16-SEP-13	245.51
9	1275	ELEANORE	NEFF	556 CACHE DRIVE	Saint Stephens	AL	36569	07-DEC-13	428.73
10	1407	FELICIA	CRUZ	643 TURNAGAIN PARKWAY	Coalburg	AL	35068	07-JAN-14	387.93
11	1440	LONNIE	MANNING	644 PARKER PLACE	Big Cove	AL	35763	16-JAN-14	377.51
12	152	LISETTE	WHITTAKER	339 NORTHPARK DRIVE	Montgomery	AL	36197	20-NOV-13	139.22
13	219	CATHI	WHITEHEAD	760 WOODCLIFF DRIVE	Huntsville	AL	35893	20-NOV-13	273.23
14	286	JEANNE	STEINER	1974 SCHUSS DRIVE	Carrollton	AL	35447	21-MAR-13	226.4
15	820	MARCELA	DUGAN	1785 DORIS PLACE	Sylacauga	AL	35150	05-AUG-13	195.34
16	1068	ELIZA	CURRIE	778 LOUDERMILK CIRCLE	Panola	AL	35477	21-OCT-13	249.92
17	1172	ADELE	PERKINS	1192 RICHARDSON VISTA ROAD	Sylacauga	AL	35150	20-DEC-13	193.9
18	1253	TRENTON	PRESLEY	1905 ANCHOR PARK CIRCLE	Montgomery	AL	36131	29-NOV-13	348.95
19	1264	MICHAELA	RICHARD	44 RASMUSSON STREET	Georgetown	AL	36521	03-DEC-13	143.96
20	188	LUANNE	GOODWIN	293 KIANA AVENUE	Pinegrove	AL	36507	12-AUG-13	202.41
21	696	ALISHA	TOMLINSON	1985 EAST 52ND AVENUE	Catherine	AL	36728	08-JUL-13	37.98
22	798	LETICIA	HEBERT	1244 DEER PARK DRIVE	Shorterville	AL	36373	30-JUL-13	41.07
23	1100	ELEANORE	SAUNDERS	820 QUARTZ AVENUE	Silver Cross	AL	36538	19-OCT-13	326.03
24	1350	LATONYA	KAY	61 LOUSSAC DRIVE	Seaboard	AL	36529	25-DEC-13	275.46
25	1439	SHANNA	PRITCHARD	1448 ARLENE DRIVE	Lavaca	AL	36911	15-JAN-14	71.98

7. One of the purchasing managers is interested in the impact of product prices on the sale of products of each brand. Write a query to display the brand name, brand type, average price of products of each brand, and total units sold of products of each brand. Even if a product has been sold more than once, its price should only be included once in the calculation of the average price. However, you must be careful because multiple products of the same brand can have the same price, and each of those products must be included in the calculation of the brand's average price.

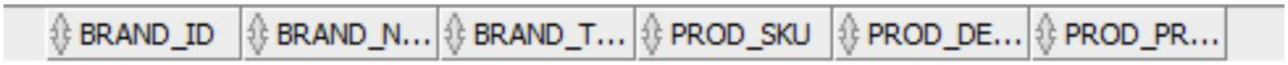
```
SELECT A.BRAND_ID, B.BRAND_NAME, B.BRAND_TYPE
,A.AVG_PROD_PRICE,B.TOT_UNIT_SOLD
FROM ((SELECT b.BRAND_ID,ROUND(AVG(p.PROD_PRICE),4) AS AVG_PROD_PRICE
FROM PRODUCT p,BRAND b
WHERE p.BRAND_ID = b.BRAND_ID
GROUP BY b.BRAND_ID) A
INNER JOIN (SELECT b1.BRAND_ID,b1.BRAND_NAME,b1.BRAND_TYPE,
SUM(l.LINE_QTY) AS TOT_UNIT_SOLD
FROM LINE l ,PRODUCT p1 ,BRAND b1
WHERE b1.BRAND_ID = p1.BRAND_ID AND p1.PROD_SKU=l.PROD_SKU
GROUP BY b1.BRAND_ID,b1.BRAND_NAME,b1.BRAND_TYPE) B
ON A.BRAND_ID=B.BRAND_ID);
```



	BRAND_ID	BRAND_NAME	BRAND_TYPE	AVG_PROD_PRICE	TOT_UNIT_SOLD
1	30	LONG HAUL	CONTRACTOR	20.1193	665
2	25	STUTTENFURST	CONTRACTOR	16.4678	401
3	28	OLDE TYME QUALITY	CONTRACTOR	18.3307	430
4	29	BUSTERS	VALUE	22.586	479
5	31	VALU-MATTE	VALUE	16.84	312
6	35	LE MODE	PREMIUM	19.2206	561
7	23	FORESTERS BEST	VALUE	20.9433	221
8	33	BINDER PRIME	PREMIUM	16.1159	413
9	27	HOME COMFORT	CONTRACTOR	21.7956	466

8. The purchasing manager is still concerned about the impact of price on sales. Write a query to display the brand name, brand type, product SKU, product description, and price of any products that are not a premium brand, but that cost more than the most expensive premium brand products.

```
SELECT b.BRAND_ID, b.BRAND_NAME,b.BRAND_TYPE,p.PROD_SKU,
p.PROD_DESCRIPT,p.PROD_PRICE
FROM BRAND b,PRODUCT p
WHERE b.BRAND_ID=p.BRAND_ID
AND b.BRAND_TYPE <> 'PREMIUM'
AND p.PROD_PRICE > ANY
( SELECT MAX(p.PROD_PRICE)
FROM PRODUCT p1,BRAND b1
WHERE p1.BRAND_ID = b1.BRAND_ID
AND b1.BRAND_TYPE = 'PREMIUM'
GROUP BY b1.BRAND_ID);
```



BRAND_ID	BRAND_N...	BRAND_T...	PROD_SKU	PROD_DE...	PROD_PR...

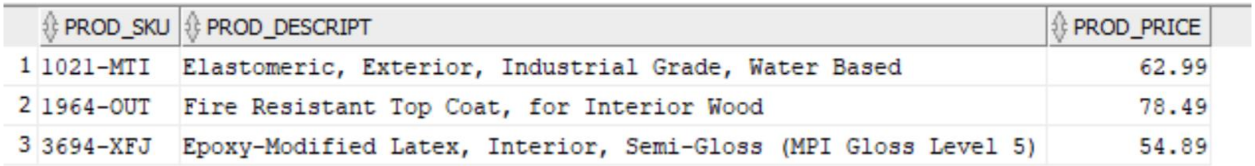
9. Using SQL descriptive statistics functions calculate the value of the following items:
    a. What are the products that have a price greater than $50?

SELECT p.PROD_SKU,p.PROD_DESCRIPT,p.PROD_PRICE

FROM PRODUCT p

WHERE p.PROD_PRICE > 50

ORDER BY 1;

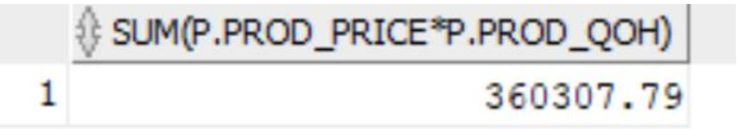


	PROD_SKU	PROD_DESCRIPT	PROD_PRICE
1	1021-MTI	Elastomeric, Exterior, Industrial Grade, Water Based	62.99
2	1964-OUT	Fire Resistant Top Coat, for Interior Wood	78.49
3	3694-XFJ	Epoxy-Modified Latex, Interior, Semi-Gloss (MPI Gloss Level 5)	54.89

    b. What is total value of our entire inventory on hand?

SELECT SUM(p.PROD_PRICE * p.PROD_QOH)

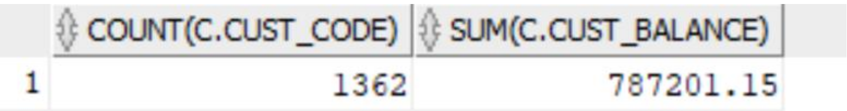
FROM PRODUCT p;



	SUM(P.PROD_PRICE*P.PROD_QOH)
1	360307.79

    c. How many customers do we presently have and what is the total of all customer balances?

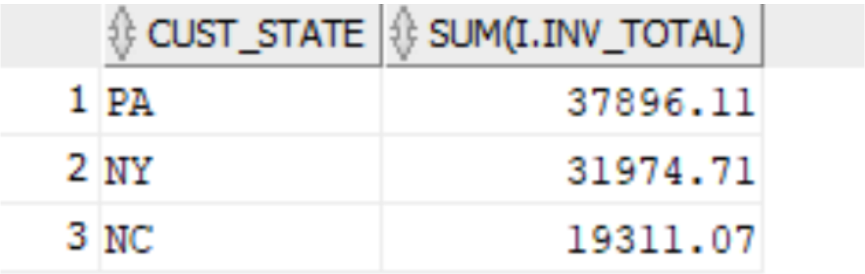
SELECT COUNT(c.CUST_CODE),SUM(c.CUST_BALANCE)
FROM CUSTOMER c;



	COUNT(C.CUST_CODE)	SUM(C.CUST_BALANCE)
1	1362	787201.15

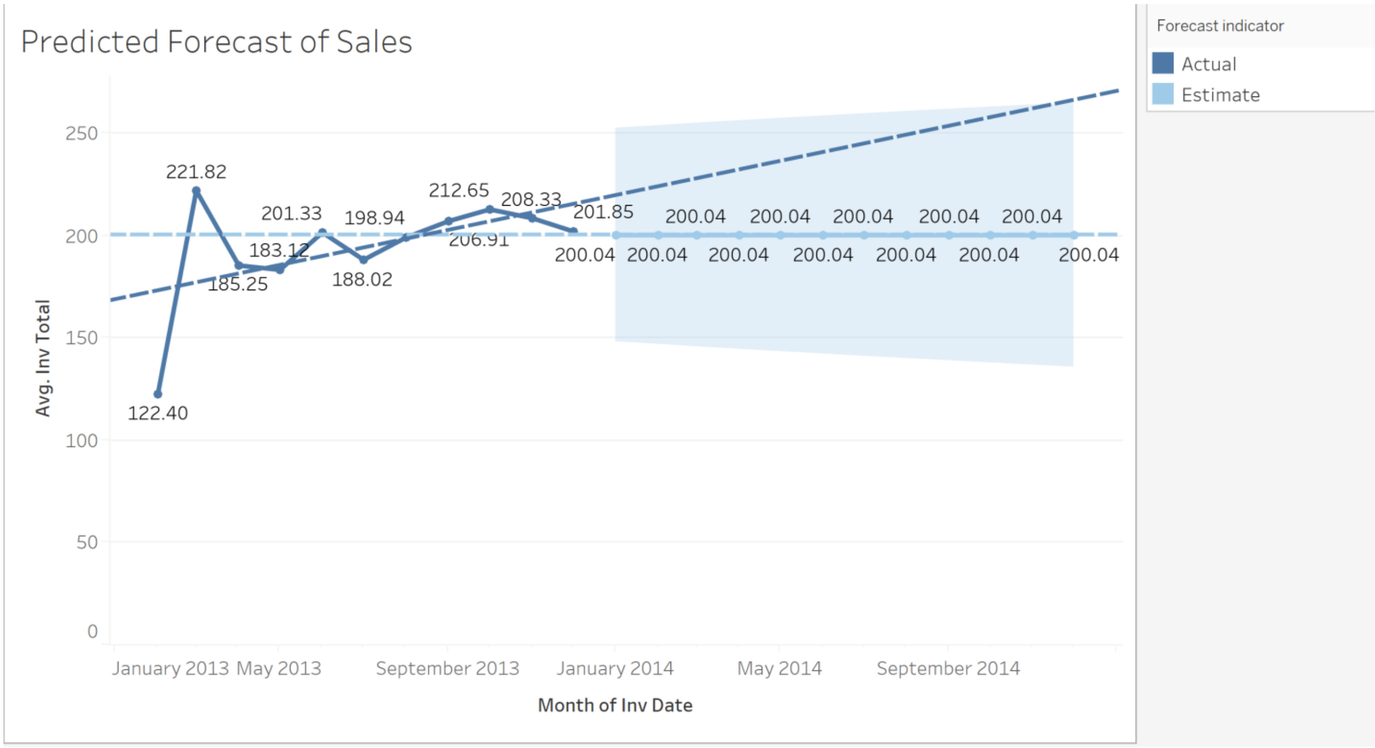
d. What are to top three states that buy the most product in dollars from the company?

```
SELECT * FROM (SELECT c.CUST_STATE, SUM(i.INV_TOTAL)

FROM CUSTOMER c LEFT JOIN INVOICE i

ON i.CUST_CODE = c.CUST_CODE

GROUP BY c.CUST_STATE

ORDER BY 2 DESC)

WHERE ROWNUM <= 3;
```



	CUST_STATE	SUM(I.INV_TOTAL)
1	PA	37896.11
2	NY	31974.71
3	NC	19311.07

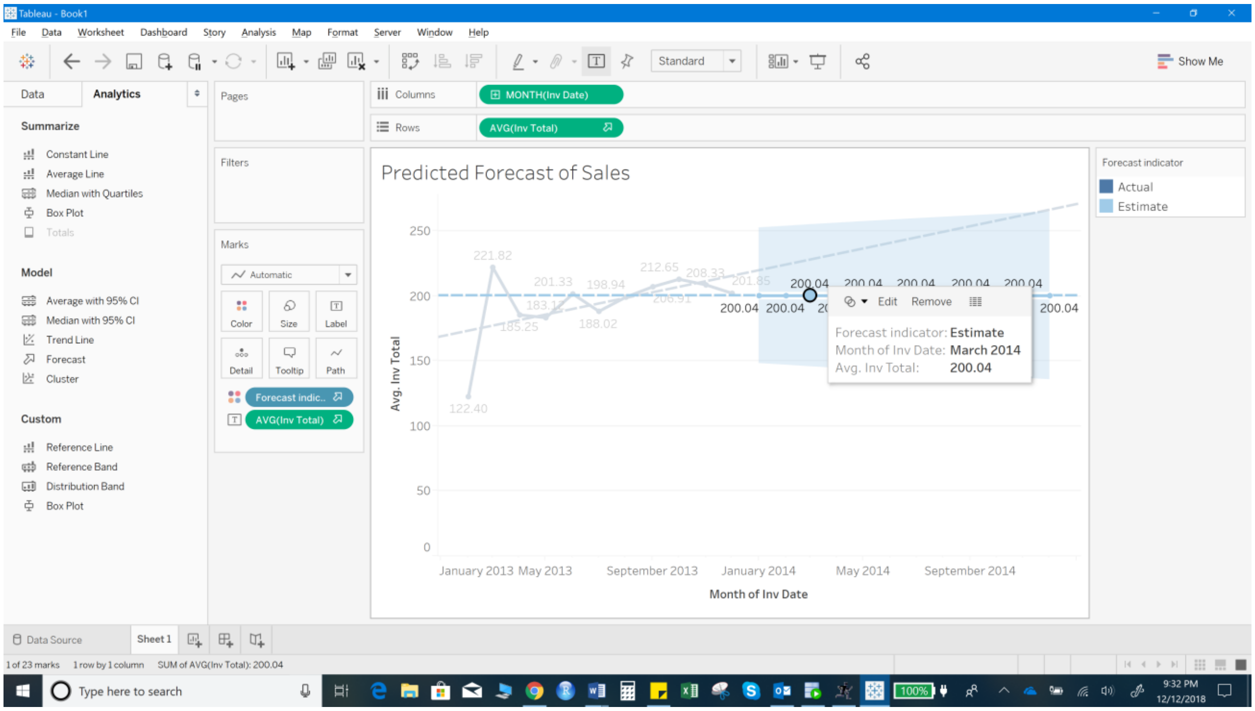
10. Using predictive statistics calculate what the predicted forecast of sales for the next year based on the INV_DATE (independent) and INV_TOTAL (dependent). Remember that you will need to convert the INV_DATE from the MS SQL Server stored date value to the expect Julian date, since numbers in MS SQL are stored as the number of days since 1/1/1900 with the fraction as the portion of a day (if you are using a different DBMS use the appropriate code for conversion.)



R-Squared: 0.306402

Siddhartha Shankar, MS-BA
NET ID: SXS180085

Average Forecasted value for the next year = $200.04 as per the screenshot below.



Forecast indicator: Estimate
Month of Inv Date: March 2014
Avg. Inv Total: 200.04

1 of 23 marks 1 row by 1 column SUM of AVG(Inv Total): 200.04

References:

https://stackoverflow.com/questions/10294284/remove-all-special-characters-from-a-string-in-r

http://rprogramming.net/connect-to-database-in-r/

https://onlinehelp.tableau.com/current/pro/desktop/en-us/forecast_create.htm

http://www.fekete.com/san/webhelp/feketeharmony/harmony_webhelp/content/html_files/Setting_up_Your_Database_for_Import/Creating_a_DSN/Creating_a_DSN_for_an_Oracle_DB.htm