1- Denormalize ABC_Retail tables

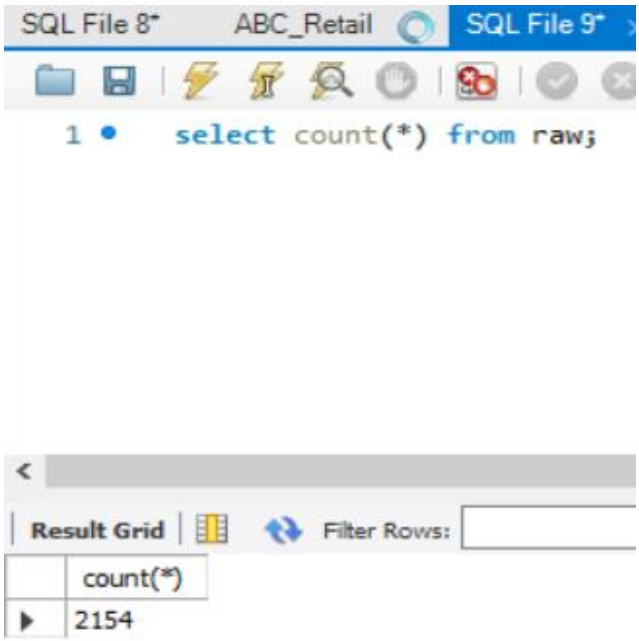
See the code file **denormalize_abc_retail.sql**

Check the result:
Table **raw** is the table that we imported the original ABC_Retail.csv 2 weeks ago.
Table **ABC_Retail** is the table which we de-normalize all data from other tables and insert into it.

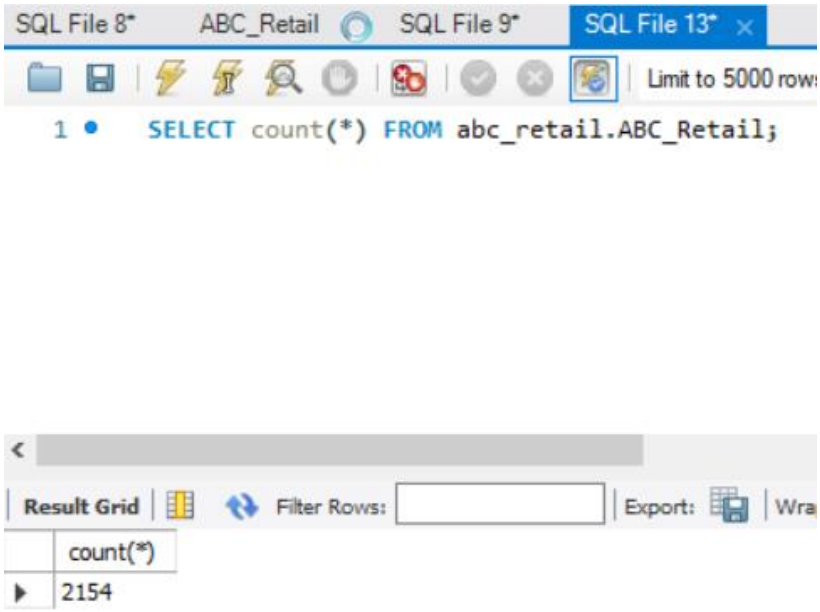
```
-- create table
CREATE TABLE abc_retail.ABC_Retail(
OrderID int,
OrderDate date default null,
Order_ShippedDate date default null,
Order_Freight numeric(20,2),
Order_ShipCity varchar(255),
Order_ShipCountry varchar(255),
Order_UnitPrice numeric(20,2),
Order_Quantity numeric(20,2),
Order_Amount numeric(20,2),
ProductName varchar(255),
Employee_LastName varchar(255),
Employee_FirstName varchar(255),
Employee_Title varchar(255),
CompanyName varchar(255),
Customer_ContactName varchar(255),
Customer_City varchar(255),
Customer_Country varchar(255),
Customer_Phone varchar(255)
);

-- denormalize tables into table ABC_Retail
INSERT INTO abc_retail.ABC_Retail
(
    OrderID, OrderDate, Order_ShippedDate, Order_Freight, Order_ShipCit
y,
    Order_ShipCountry, Order_UnitPrice, Order_Quantity, Order_Amount,
    ProductName, Employee_LastName, Employee_FirstName, Employee_Title,
    CompanyName, Customer_ContactName, Customer_City, Customer_Country,
    Customer_Phone
)
(
    SELECT DISTINCT
```

```
        o.order_id, o.order_date, o.order_shippeddate, o.order_freight,
        o.order_shipcity, o.order_shipcountry,
        opp.order_unitprice, opp.order_quantity, opp.order_amount,
        opp.product_name,
        e.employee_lastname, e.employee_firstname, e.employee_title,
        cocu.company_name,
        cocu.customer_contactname, cocu.customer_city, cocu.customer_co
untry,
        cocu.customer_phone
    FROM abc_retail.orders AS o
    JOIN abc_retail.employees as e
    ON o.employee_id = e.employee_id
    JOIN
    (
        SELECT DISTINCT
            cu.customer_id,
            co.company_name,
            cu.customer_contactname, cu.customer_city, cu.customer_coun
try,
            cu.customer_phone
        FROM abc_retail.companys AS co
        JOIN abc_retail.customers AS cu
        ON co.company_id = cu.company_id
    ) AS cocu
    ON o.customer_id = cocu.customer_id
    JOIN
    (
        SELECT DISTINCT
            op.order_id, op.order_unitprice, op.order_quantity, op.orde
r_amount,
            p.product_name
        FROM abc_retail.order_products AS op
        JOIN abc_retail.products AS p
        ON op.product_id = p.product_id
    ) AS opp
    ON o.order_id = opp.order_id
);
```

Check number of data in table **raw**.



Check number of data in table **ABC_Retail**.



Data in table **raw**.

ABC_Retail | SQL File 9* | SQL File 13*

Limit to 100 rows

```
1 • select * from abc_retail.raw order by order_id limit 50;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content: | Fetch rows:

order_id	order_date	order_shippeddate	order_freight	order_shipcity	order_shipcountry	order_unitprice	order_quantity	ord
1	2016-07-04	2016-07-16	32.38	Reims	France	9.80	10.00	98.0
1	2016-07-04	2016-07-16	32.38	Reims	France	34.80	5.00	174.
2	2016-07-05	2016-07-10	11.61	Münster	Germany	18.60	9.00	167.
2	2016-07-05	2016-07-10	11.61	Münster	Germany	42.40	40.00	1696
3	2016-07-08	2016-07-12	65.83	Rio de Janeiro	Brazil	7.70	10.00	77.0
3	2016-07-08	2016-07-12	65.83	Rio de Janeiro	Brazil	42.40	35.00	148
3	2016-07-08	2016-07-12	65.83	Rio de Janeiro	Brazil	16.80	15.00	252.
4	2016-07-08	2016-07-15	41.34	Lyon	France	16.80	6.00	100.
4	2016-07-08	2016-07-15	41.34	Lyon	France	15.60	15.00	234.
4	2016-07-08	2016-07-15	41.34	Lyon	France	16.80	20.00	336.
5	2016-07-09	2016-07-11	51.30	Charleroi	Belgium	64.80	40.00	259

Data in table **ABC_Retail**.

ABC_Retail | SQL File 9*

Limit to 100 rows

```
1 • SELECT * FROM abc_retail.ABC_Retail order by OrderID limit 50;
2
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content: | Fetch rows:

OrderID	OrderDate	Order_ShippedDate	Order_Freight	Order_ShipCity	Order_ShipCountry	Order_UnitPrice	Order_Quantity
1	2016-07-04	2016-07-16	32.38	Reims	France	34.80	5.00
1	2016-07-04	2016-07-16	32.38	Reims	France	9.80	10.00
2	2016-07-05	2016-07-10	11.61	Münster	Germany	42.40	40.00
2	2016-07-05	2016-07-10	11.61	Münster	Germany	18.60	9.00
3	2016-07-08	2016-07-12	65.83	Rio de Janeiro	Brazil	7.70	10.00
3	2016-07-08	2016-07-12	65.83	Rio de Janeiro	Brazil	16.80	15.00
3	2016-07-08	2016-07-12	65.83	Rio de Janeiro	Brazil	42.40	35.00
4	2016-07-08	2016-07-15	41.34	Lyon	France	16.80	6.00
4	2016-07-08	2016-07-15	41.34	Lyon	France	16.80	20.00
4	2016-07-08	2016-07-15	41.34	Lyon	France	15.60	15.00
5	2016-07-09	2016-07-11	51.30	Charleroi	Belgium	27.20	40.00

2- Create MyCube table from the denormalized table

Since **select ··· into** is not valid in our MySQL version, detail reasons could be found at https://stackoverflow.com/questions/2949653/select-into-and-undeclared-variable-error

We just use **create** then **insert into ··· select ···**

```
-- create table
CREATE TABLE abc_retail.MyCube(
    ThisYear year default null,
    Region varchar(255),
    Product varchar(255),
    Sales numeric(20,2)
);

-- Create MyCube table from the denormalized table
INSERT INTO abc_retail.MyCube
(
    ThisYear, Region, Product, Sales
)
SELECT
    year(OrderDate) as ThisYear
    ,Order_ShipCountry as Region
    ,ProductName as Product
    ,Order_Amount as Sales
FROM
    abc_retail.ABC_Retail
WHERE
    Order_ShipCountry in ('USA','Canada','UK')
    and ProductName in ('Chai','Tofu','Chocolade');
```

Check the result:

```
1 • SELECT * FROM abc_retail.MyCube;
```

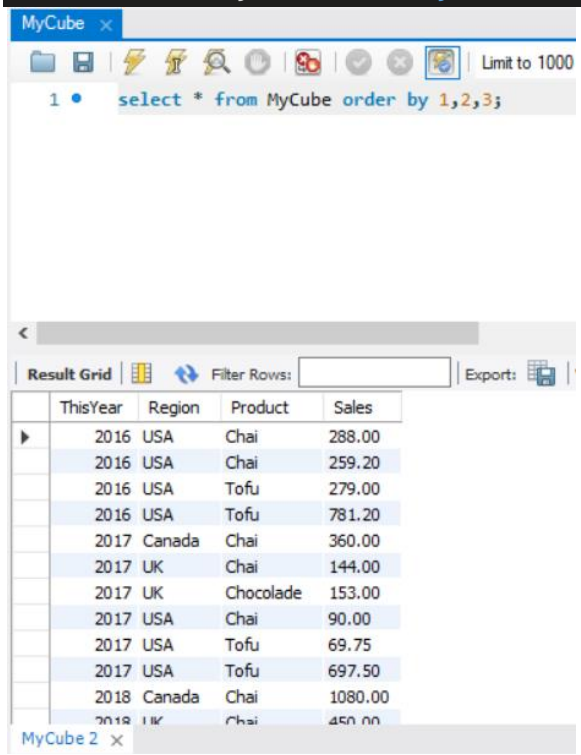
Result Grid | Filter Rows: | Expo

ThisYear	Region	Product	Sales
2017	UK	Chocolade	153.00
2018	Canada	Chai	1080.00
2018	UK	Chai	450.00
2017	USA	Tofu	69.75
2018	USA	Chai	144.00
2016	USA	Tofu	279.00
2016	USA	Chai	288.00

3- Run all the attached OLAP Operators against the newly created MyCube table

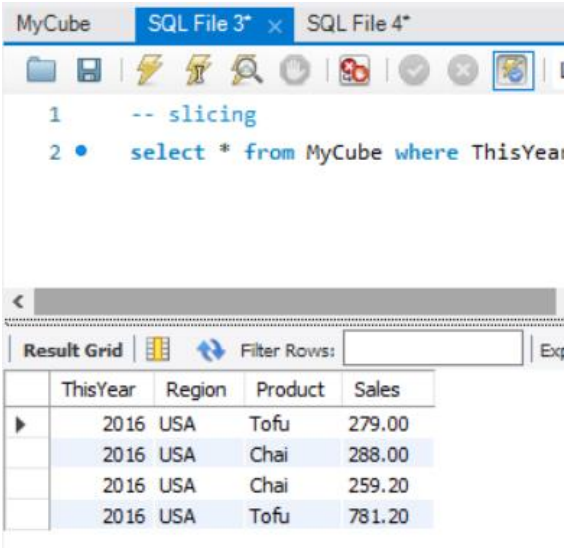
The following OLAP operators could be run in our MySQL database.
And other operations like the **PIVOT** operators could not be run on MySQL.

```
-- table
select * from MyCube order by 1,2,3
```



ThisYear	Region	Product	Sales
2016	USA	Chai	288.00
2016	USA	Chai	259.20
2016	USA	Tofu	279.00
2016	USA	Tofu	781.20
2017	Canada	Chai	360.00
2017	UK	Chai	144.00
2017	UK	Chocolade	153.00
2017	USA	Chai	90.00
2017	USA	Tofu	69.75
2017	USA	Tofu	697.50
2018	Canada	Chai	1080.00
2018	UK	Chai	450.00

```
-- slicing
select * from MyCube where ThisYear=2016
```



ThisYear	Region	Product	Sales
2016	USA	Tofu	279.00
2016	USA	Chai	288.00
2016	USA	Chai	259.20
2016	USA	Tofu	781.20

```sql
-- dicing
select * from MyCube where ThisYear=2016 and region='USA'
```

MyCube | SQL File 3* | SQL File 4*

Limit to 100 rows

```sql
-- dicing
select * from MyCube where ThisYear=2016 and region='USA'
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

ThisYear	Region	Product	Sales
2016	USA	Tofu	279.00
2016	USA	Chai	288.00
2016	USA	Chai	259.20
2016	USA	Tofu	781.20

```sql
-- group by with rollup
SELECT ThisYear, Region, Product, SUM(Sales)as TotalSales-
-, GROUPING(ThisYear) AS 'Grouping'
FROM MyCube
GROUP BY ThisYear, Region, Product with rollup
ORDER BY 1,2,3
```

Result Grid | Filter Rows:

ThisYear	Region	Product	TotalSales
NULL	NULL	NULL	7860.90
2016	NULL	NULL	1607.40
2016	USA	NULL	1607.40
2016	USA	Chai	547.20
2016	USA	Tofu	1060.20
2017	NULL	NULL	1514.25
2017	Canada	NULL	360.00
2017	Canada	Chai	360.00
2017	UK	NULL	297.00
2017	UK	Chai	144.00
2017	UK	Chocol...	153.00
2017	USA	NULL	857.25
2017	USA	Chai	90.00
2017	USA	Tofu	767.25
NULL	NULL	NULL	[illegible]

Result 16

ThisYear	Region	Product	TotalSales	Grouping
NULL	NULL	NULL	7860.90	1
2016	NULL	NULL	1607.40	0
2016	USA	NULL	1607.40	0
2016	USA	Chai	547.20	0
2016	USA	Tofu	1060.20	0
2017	NULL	NULL	1514.25	0
2017	Canada	NULL	360.00	0
2017	Canada	Chai	360.00	0
2017	UK	NULL	297.00	0
2017	UK	Chai	144.00	0
2017	UK	Chocol...	153.00	0
2017	USA	NULL	857.25	0
2017	USA	Chai	90.00	0
2017	USA	Tofu	767.25	0
NULL	NULL	NULL	[illegible]	0

```sql
-- group by with cube
SELECT ThisQuarter, Region, Product, SUM(Sales)
as TotalSales--, GROUPING(ThisQuarter) AS 'Grouping'
FROM MyCube
GROUP BY ThisQuarter, Region, Product with cube
ORDER BY 1,2,3
```

Could be write as:

```sql
(
    SELECT ThisYear, Region, Product, SUM(Sales)as TotalSales-
-, GROUPING(ThisYear) AS 'Grouping'
    FROM MyCube
    GROUP BY ThisYear, Region, Product with rollup
    UNION
    SELECT ThisYear, Region, Product, SUM(Sales)as TotalSales-
-, GROUPING(ThisYear) AS 'Grouping'
    FROM MyCube
    GROUP BY ThisYear, Product, Region with rollup
    UNION
    SELECT ThisYear, Region, Product, SUM(Sales)as TotalSales-
-, GROUPING(ThisYear) AS 'Grouping'
    FROM MyCube
    GROUP BY Region, ThisYear, Product with rollup
    UNION
    SELECT ThisYear, Region, Product, SUM(Sales)as TotalSales-
-, GROUPING(ThisYear) AS 'Grouping'
    FROM MyCube
    GROUP BY Region, Product, ThisYear with rollup
    UNION
    SELECT ThisYear, Region, Product, SUM(Sales)as TotalSales-
-, GROUPING(ThisYear) AS 'Grouping'
    FROM MyCube
    GROUP BY Product, ThisYear, Region with rollup
    UNION
    SELECT ThisYear, Region, Product, SUM(Sales)as TotalSales-
-, GROUPING(ThisYear) AS 'Grouping'
    FROM MyCube
    GROUP BY Product, Region, ThisYear with rollup
)
ORDER BY 1,2,3;
```

ThisYear	Region	Product	TotalSales
2016	USA	NULL	1607.40
2016	USA	Chai	547.20
2016	USA	Tofu	1060.20
2017	NULL	NULL	1514.25
2017	NULL	Chai	594.00
2017	NULL	Chocolade	153.00
2017	NULL	Tofu	767.25
2017	Canada	NULL	360.00
2017	Canada	Chai	360.00
2017	UK	NULL	297.00
2017	UK	Chai	144.00
2017	UK	Chocolade	153.00
2017	USA	NULL	857.25
2017	USA	Chai	90.00

ThisYear	Region	Product	TotalSales	Grouping
NULL	NULL	NULL	7860.90	1
NULL	NULL	Chai	5857.20	1
NULL	NULL	Chocolade	153.00	1
NULL	NULL	Tofu	1850.70	1
NULL	Canada	NULL	1440.00	1
NULL	Canada	Chai	1440.00	1
NULL	UK	NULL	1467.00	1
NULL	UK	Chai	1314.00	1
NULL	UK	Chocolade	153.00	1
NULL	USA	NULL	4953.90	1
NULL	USA	Chai	3103.20	1
NULL	USA	Tofu	1850.70	1
2016	NULL	NULL	1607.40	0
2016	NULL	Chai	547.20	0
2016	NULL	Tofu	1060.20	0
2016	USA	NULL	1607.40	0
2016	USA	Chai	547.20	0
2016	USA	Tofu	1060.20	0
2017	NULL	NULL	1514.25	0
2017	NULL	Chai	594.00	0
2017	NULL	Chocolade	153.00	0
2017	NULL	Tofu	767.25	0
2017	Canada	NULL	360.00	0

```
-- group by grouping sets
SELECT ThisQuarter, Region, SUM(Sales) as TotalSales
FROM MyCube
GROUP BY GROUPING SETS ((ThisQuarter), (Region))
ORDER BY 1,2
```

Is the same as:

```
--
SELECT ThisYear, NULL as Region, SUM(Sales) as TotalSales FROM MyCube G
ROUP BY ThisYear
UNION ALL
SELECT NULL, Region, SUM(Sales) as TotalSales FROM MyCube GROUP BY Regi
on
ORDER BY 1,2
```

ThisYear	Region	TotalSales
NULL	Canada	1440.00
NULL	UK	1467.00
NULL	USA	4953.90
2016	NULL	1607.40
2017	NULL	1514.25
2018	NULL	4739.25

```sql
-- Ranking
SELECT
    Product, Sales
    , RANK() OVER (ORDER BY Sales ASC) as RANK_SALES
    , DENSE_RANK() OVER (ORDER BY Sales ASC) as DENSE_RANK_SALES
    , PERCENT_RANK() OVER (ORDER BY Sales ASC) as PERC_RANK_SALES
    , CUME_DIST() OVER (ORDER BY Sales ASC) as CUM_DIST_SALES
FROM
    MyCube
ORDER BY
    RANK_SALES ASC
```

Product	Sales	RANK_SALES	DENSE_RANK_SALES	PERC_RANK_SALES	CUM_DIST_SALES
Tofu	23.25	1	1	0	0.05555555555555555
Tofu	69.75	2	2	0.058823529411764705	0.1111111111111111
Chai	72.00	3	3	0.11764705882352941	0.16666666666666666
Chai	90.00	4	4	0.17647058823529413	0.2222222222222222
Chai	144.00	5	5	0.23529411764705882	0.3333333333333333
Chai	144.00	5	5	0.23529411764705882	0.3333333333333333
Chocolade	153.00	7	6	0.35294117647058826	0.3888888888888889
Chai	259.20	8	7	0.4117647058823529	0.4444444444444444
Tofu	279.00	9	8	0.47058823529411764	0.5
Chai	288.00	10	9	0.5294117647058824	0.5555555555555556
Chai	360.00	11	10	0.5882352941176471	0.6111111111111112
Chai	450.00	12	11	0.6470588235294118	0.6666666666666666

Result 3

```sql
-- Windowing
SELECT
    ThisYear, Region, Sales
    , AVG(Sales) OVER (PARTITION BY Region ORDER BY ThisYear) AS Sales_
Avg
FROM
    MyCube
ORDER BY
    Region, ThisYear, Sales_Avg
```

ThisYear	Region	Sales	Sales_Avg
2017	Canada	360.00	360.000000
2018	Canada	1080.00	720.000000
2017	UK	153.00	148.500000
2017	UK	144.00	148.500000
2018	UK	450.00	366.750000
2018		720.00	366.750000
2016	USA	781.20	401.850000
2016	USA	259.20	401.850000
2016	USA	288.00	401.850000
2016	USA	279.00	401.850000
2017	USA	69.75	352.092857
2017	USA	697.50	352.092857
2017	USA	90.00	352.092857
2018	USA	144.00	412.825000

```sql
-- Windowing
SELECT
    ThisYear, Region, Sales
    , AVG(Sales) OVER (PARTITION BY Region ORDER BY ThisYear ROWS BETWE
EN 1 PRECEDING AND 1 FOLLOWING) AS Sales_Avg
FROM
    MyCube
ORDER BY
    Region, ThisYear, Sales_Avg
```

ThisYear	Region	Sales	Sales_Avg
2017	Canada	360.00	720.000000
2018	Canada	1080.00	720.000000
2017	UK	153.00	148.500000
2017	UK	44.00	249.000000
2018	UK	450.00	438.000000
2018	UK	720.00	585.000000
2016	USA	288.00	275.400000
2016	USA	279.00	283.500000
2016	USA	781.20	370.050000
2016	USA	259.20	442.800000
2017	USA	697.50	285.750000
2017	USA	90.00	310.500000
2017	USA	69.75	516.150000
2018	USA	144.00	85.750000

4- If your DBMS does not support PIVOT, write a SQL script to produce the same result

Our DBMS is MySQL server.

```
-- pivot query
select
    Product, Q1, Q2, Q3, Q4
from
    MyCube PIVOT(SUM(Sales) FOR ThisQuarter IN (Q1,Q2,Q3,Q4)) AS P
```

Could be write as:

```
-- pivot query
select
    Product,
    SUM( IF(ThisYear=2016,Sales,0) ) As `2016`,
    SUM( IF(ThisYear=2017,Sales,0) ) As `2017`,
    SUM( IF(ThisYear=2018,Sales,0) ) As `2018`,
    SUM(Sales) As 'ALL'
from MyCube
group by Product with rollup;
```

Product	2016	2017	2018	ALL
Chai	547.20	594.00	4716.00	5857.20
Chocolade	0.00	153.00	0.00	153.00
Tofu	1060.20	767.25	23.25	1850.70
NULL	1607.40	1514.25	4739.25	7860.90

```
-- pivot query
SELECT Product, Region, Q1, Q2, Q3, Q4
FROM
(SELECT Product, Region, ThisQuarter, Sales FROM MyCube) AS p
PIVOT
(sum(Sales) FOR ThisQuarter IN (Q1,Q2,Q3,Q4)) AS pvt
```

Could be write as:

```
SELECT
    Product,
    Region,
    sum( IF(ThisYear=2016,Sales,0) ) As `2016`,
    sum( IF(ThisYear=2017,Sales,0) ) As `2017`,
    sum( IF(ThisYear=2018,Sales,0) ) As `2018` ,
    sum(Sales) As 'ALL'
```

```sql
FROM
(SELECT Product, Region, ThisYear, Sales FROM MyCube) AS p
GROUP BY Product, Region WITH ROLLUP;
```

Product	Region	2016	2017	2018	ALL
Chai	Canada	0.00	360.00	1080.00	1440.00
Chai	UK	0.00	144.00	1170.00	1314.00
Chai	USA	547.20	90.00	2466.00	3103.20
Chai	NULL	547.20	594.00	4716.00	5857.20
Chocolade	UK	0.00	153.00	0.00	153.00
Chocolade	NULL	0.00	153.00	0.00	153.00
Tofu	USA	1060.20	767.25	23.25	1850.70
Tofu	NULL	1060.20	767.25	23.25	1850.70
NULL	NULL	1607.40	1514.25	4739.25	7860.90