SQL Joins – Cartesian Product

Beware the Cartesian Product !!

Product = one table multiplied by another table

The JOIN often creates a product, then selects rows from the product where the keys match

René DesCartes

SQL Joins – Cartesian Product

Beware the Cartesian Product !!

René DesCartes

Product = one table multiplied by another table

The JOIN often creates a product, then selects rows from the product where the keys match

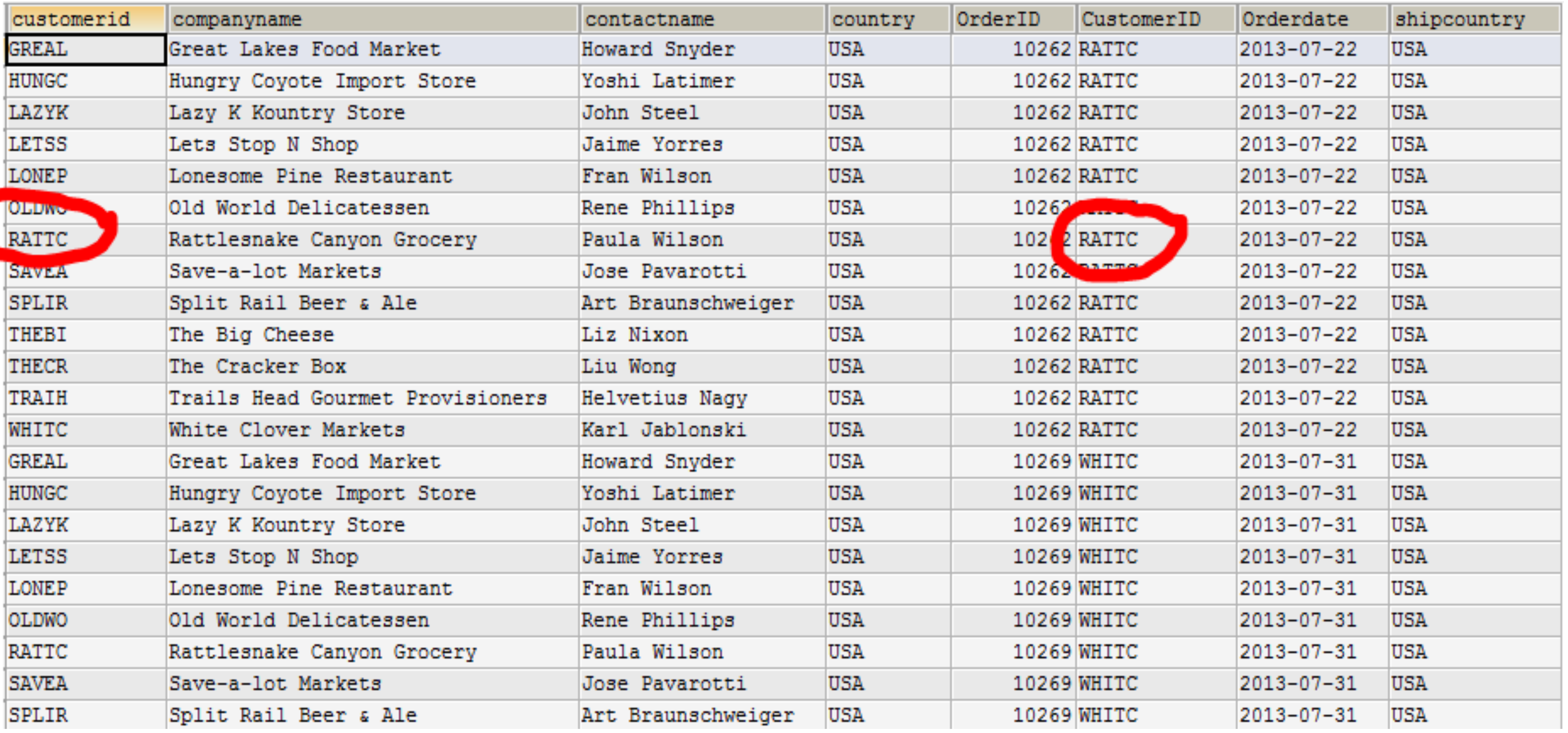
SQL Joins – Cartesian Product

For example, let's join Orders to Employees

- Orders has 14 columns, 830 rows
- Employees has 17 columns, 9 rows
- The Cartesian product has 31 (14+17) columns, and 7470 (830 * 9) rows – most of which are meaningless

SQL Joins - Cartesian Product

Cartesian Product from an unqualified join:



customerid	companyname	contactname	country	OrderID	CustomerID	Orderdate	shipcountry
GREAL	Great Lakes Food Market	Howard Snyder	USA	10262	RATTC	2013-07-22	USA
HUNGC	Hungry Coyote Import Store	Yoshi Latimer	USA	10262	RATTC	2013-07-22	USA
LAZYK	Lazy K Kountry Store	John Steel	USA	10262	RATTC	2013-07-22	USA
LETSS	Lets Stop N Shop	Jaime Yorres	USA	10262	RATTC	2013-07-22	USA
LONEP	Lonesome Pine Restaurant	Fran Wilson	USA	10262	RATTC	2013-07-22	USA
OLDWO	Old World Delicatessen	Rene Phillips	USA	10262	RATTC	2013-07-22	USA
RATTC	Rattlesnake Canyon Grocery	Paula Wilson	USA	10262	RATTC	2013-07-22	USA
SAVEA	Save-a-lot Markets	Jose Pavarotti	USA	10262	RATTC	2013-07-22	USA
SPLIR	Split Rail Beer & Ale	Art Braunschweiger	USA	10262	RATTC	2013-07-22	USA
THEBI	The Big Cheese	Liz Nixon	USA	10262	RATTC	2013-07-22	USA
THECR	The Cracker Box	Liu Wong	USA	10262	RATTC	2013-07-22	USA
TRAIH	Trails Head Gourmet Provisioners	Helvetius Nagy	USA	10262	RATTC	2013-07-22	USA
WHITC	White Clover Markets	Karl Jablonski	USA	10262	RATTC	2013-07-22	USA
GREAL	Great Lakes Food Market	Howard Snyder	USA	10269	WHITC	2013-07-31	USA
HUNGC	Hungry Coyote Import Store	Yoshi Latimer	USA	10269	WHITC	2013-07-31	USA
LAZYK	Lazy K Kountry Store	John Steel	USA	10269	WHITC	2013-07-31	USA
LETSS	Lets Stop N Shop	Jaime Yorres	USA	10269	WHITC	2013-07-31	USA
LONEP	Lonesome Pine Restaurant	Fran Wilson	USA	10269	WHITC	2013-07-31	USA
OLDWO	Old World Delicatessen	Rene Phillips	USA	10269	WHITC	2013-07-31	USA
RATTC	Rattlesnake Canyon Grocery	Paula Wilson	USA	10269	WHITC	2013-07-31	USA
SAVEA	Save-a-lot Markets	Jose Pavarotti	USA	10269	WHITC	2013-07-31	USA
SPLIR	Split Rail Beer & Ale	Art Braunschweiger	USA	10269	WHITC	2013-07-31	USA

SQL Joins - Cartesian Product

Cartesian Product

- SQL will go through the Cartesian Product (which is an INTERIM answer set) row-by-row, and select only those rows where the EmployeeID from Employees is equal to the EmployeeID from Orders
- Therefore, we must include the JOIN or WHERE clause that describes this condition

Failure to fully qualify a JOIN operation with a JOIN/WHERE clause that matches all necessary keys will cause your answer set to include a Cartesian Product (which is mostly meaningless)

SQL Joins - Cartesian Product

Cartesian Product Error Example

Consider a three-way join getting data from 3 tables:

Employees, Orders, and OrderDetails

“Provide a list of all employees and the total dollar value of each employee’s orders.”

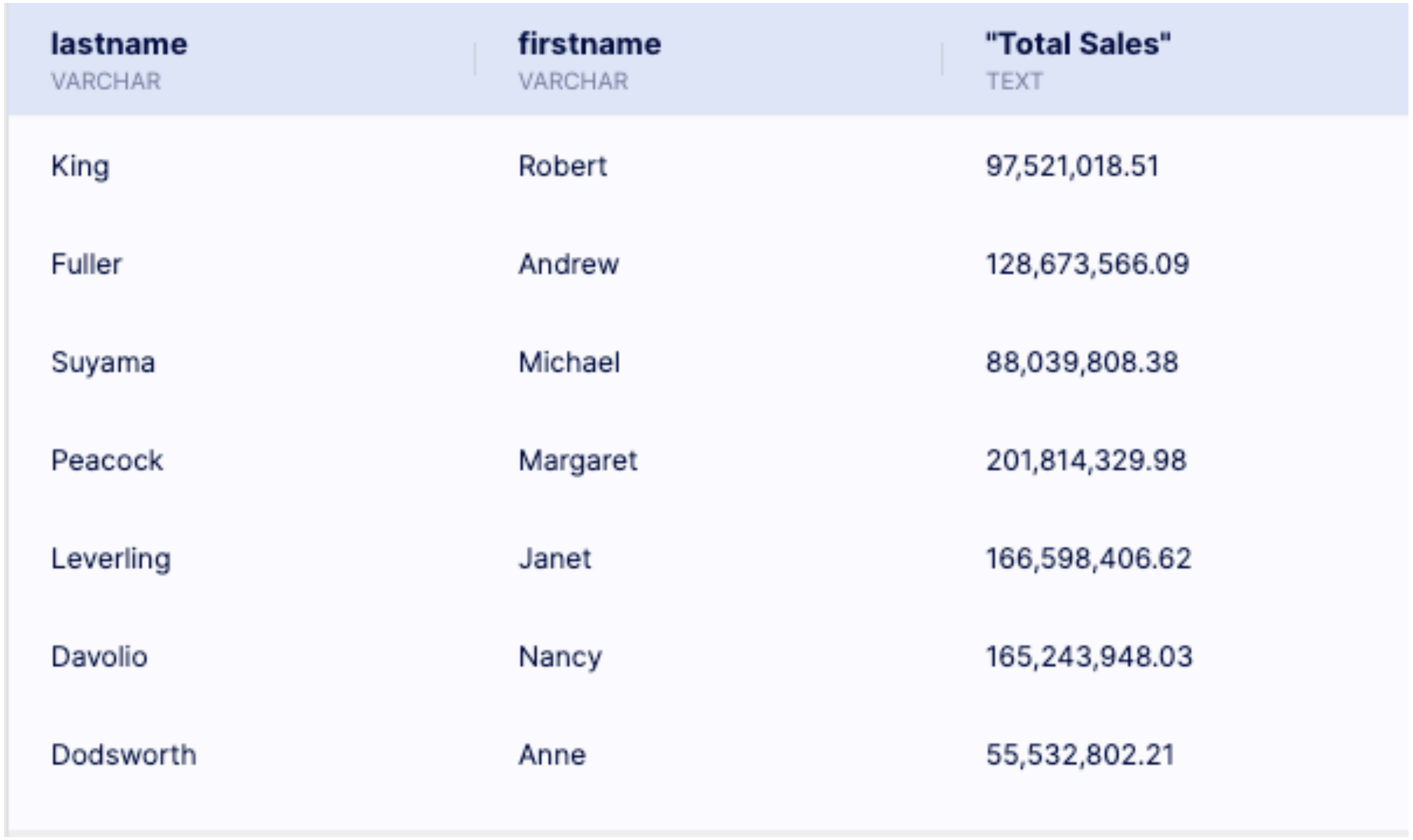
SQL Joins - Cartesian Product

```
SELECT LastName, Firstname,
        to_char(sum(unitprice * quantity),'999,999,999.99') as "Total Sales"
   from "alanparadise/nw"."employees" E,
        "alanparadise/nw"."orders" O,
        "alanparadise/nw"."orderdetails" D
   where E.employeeid  =  O.employeeid
GROUP BY LastName, FirstName
```

Three tables;
only ONE
JOIN condition

SQL Joins - Cartesian Product

At first glance, the answer set might look reasonable, but it is very wrong !!



lastname VARCHAR	firstname VARCHAR	"Total Sales" TEXT
King	Robert	97,521,018.51
Fuller	Andrew	128,673,566.09
Suyama	Michael	88,039,808.38
Peacock	Margaret	201,814,329.98
Leverling	Janet	166,598,406.62
Davolio	Nancy	165,243,948.03
Dodsworth	Anne	55,532,802.21

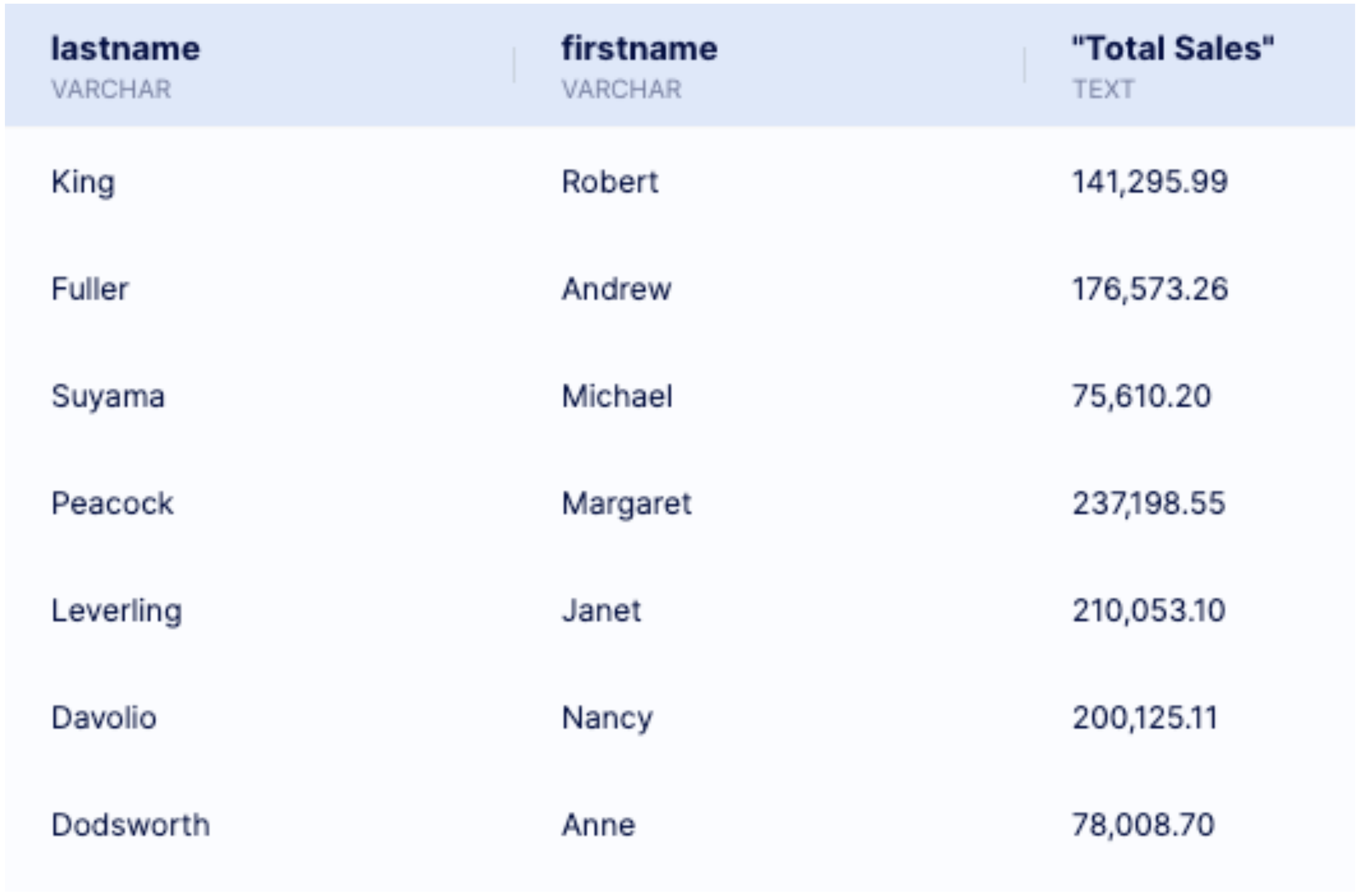
SQL Joins - Cartesian Product

```
SELECT LastName, Firstname,
        to_char(sum(unitprice * quantity),'999,999,999.99') as "Total Sales"
   from "alanparadise/nw"."employees" E,
        "alanparadise/nw"."orders" O,
        "alanparadise/nw"."orderdetails" D
   where E.employeeid  =  O.employeeid
     and O.orderidid  =  D.orderid
GROUP BY LastName, FirstName
```

Three tables;
TWO
JOIN conditions

SQL Joins - Cartesian Product

A more reasonable answer !!



lastname VARCHAR	firstname VARCHAR	"Total Sales" TEXT
King	Robert	141,295.99
Fuller	Andrew	176,573.26
Suyama	Michael	75,610.20
Peacock	Margaret	237,198.55
Leverling	Janet	210,053.10
Davolio	Nancy	200,125.11
Dodsworth	Anne	78,008.70