Nhan Vo

Amir Hammound

Setha Proeung

**#1 BASIC**

**SQL Queries:**

SELECT CUSTOMERNAME, ORDERS.ORDERNUMBER, ORDERDATE, PRODUCTNAME FROM CUSTOMERS INNER JOIN ORDERS

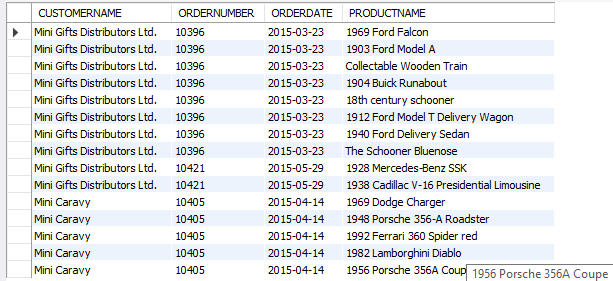
USING (CUSTOMERNUMBER)

INNER JOIN ORDERDETAILS ON ORDERS.ORDERNUMBER=ORDERDETAILS.ORDERNUMBER

INNER JOIN PRODUCTS ON PRODUCTS.PRODUCTCODE=ORDERDETAILS.PRODUCTCODE

WHERE CUSTOMERNAME LIKE '%Mini%' AND ORDERDATE > '2015-03-15';

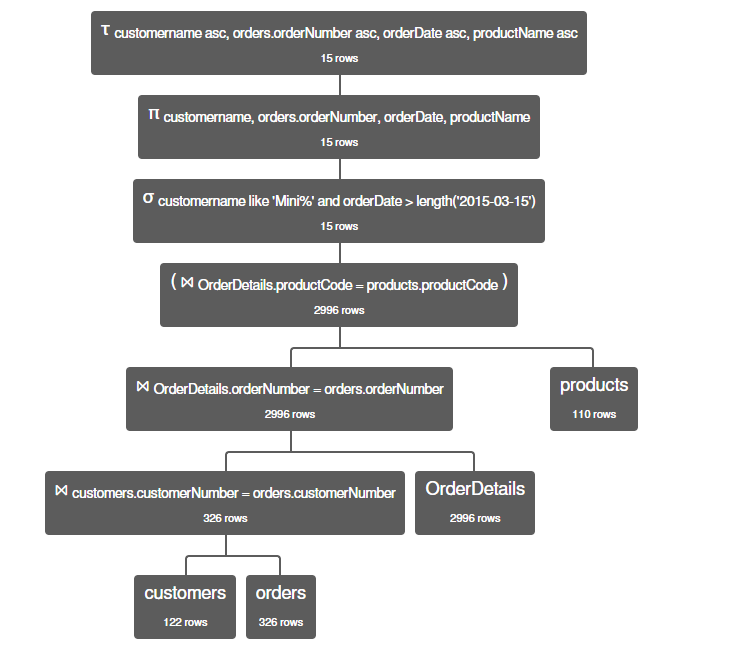
**Image:**



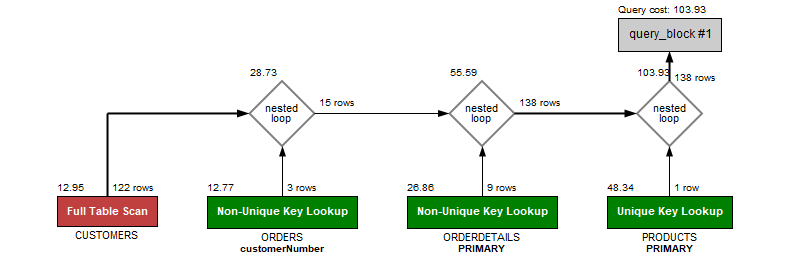
**Relational Algebra:**

τcustomername, orders.orderNumber, orderDate, productName πcustomername, orders.orderNumber, orderDate, productName σcustomername like 'Mini%' ∧ orderDate > date('2015-03-15') (customers ⨝ customers.customerNumber = orders.customerNumber orders ⨝ OrderDetails.orderNumber = orders.orderNumber OrderDetails ⨝ OrderDetails.productCode = products.productCode products)

**Image:**



**Execution (SQl visual explain):**



**Discussion:**

We do not see any difference in the number of rows. However, the relational algebra does not provide any specific row output so it is hard to track for minor mistakes.

**#2 Inline Query**

**SQL Queries:**

SELECT CUSTOMERNAME, ORDERNUMBER, ORDERDATE, PRODUCTNAME FROM

(SELECT CUSTOMERNUMBER, CUSTOMERNAME FROM CUSTOMERS WHERE CUSTOMERNAME LIKE '%Mini%') C1

INNER JOIN

(SELECT ORDERS.CUSTOMERNUMBER, ORDERS.ORDERNUMBER, ORDERDATE, PRODUCTNAME FROM ORDERS

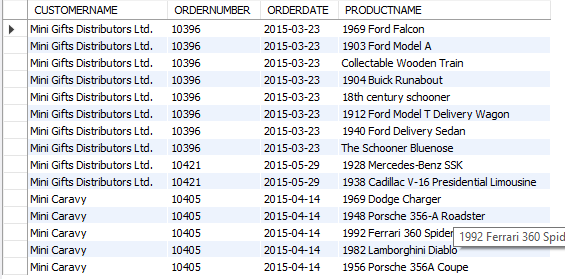
INNER JOIN ORDERDETAILS USING(ORDERNUMBER)

INNER JOIN PRODUCTS ON PRODUCTS.PRODUCTCODE =ORDERDETAILS.PRODUCTCODE

WHERE ORDERDATE > '2015-03-15') OD

ON C1.CUSTOMERNUMBER= OD.CUSTOMERNUMBER;

Image:



**Relational Algebra:**

τcustomername, orders.orderNumber, orderDate, productName

πcustomername, orders.orderNumber, orderDate, productName

σcustomername like 'Mini%'

(customers ⨝ customers.customerNumber = orders.customerNumber

orders ⨝ OrderDetails.orderNumber = orders.orderNumber

OrderDetails ⨝ OrderDetails.productCode = products.productCode

products)

∩

τcustomername, orders.orderNumber, orderDate, productName

πcustomername, orders.orderNumber, orderDate, productName

σ orderDate> date('2015-03-15')

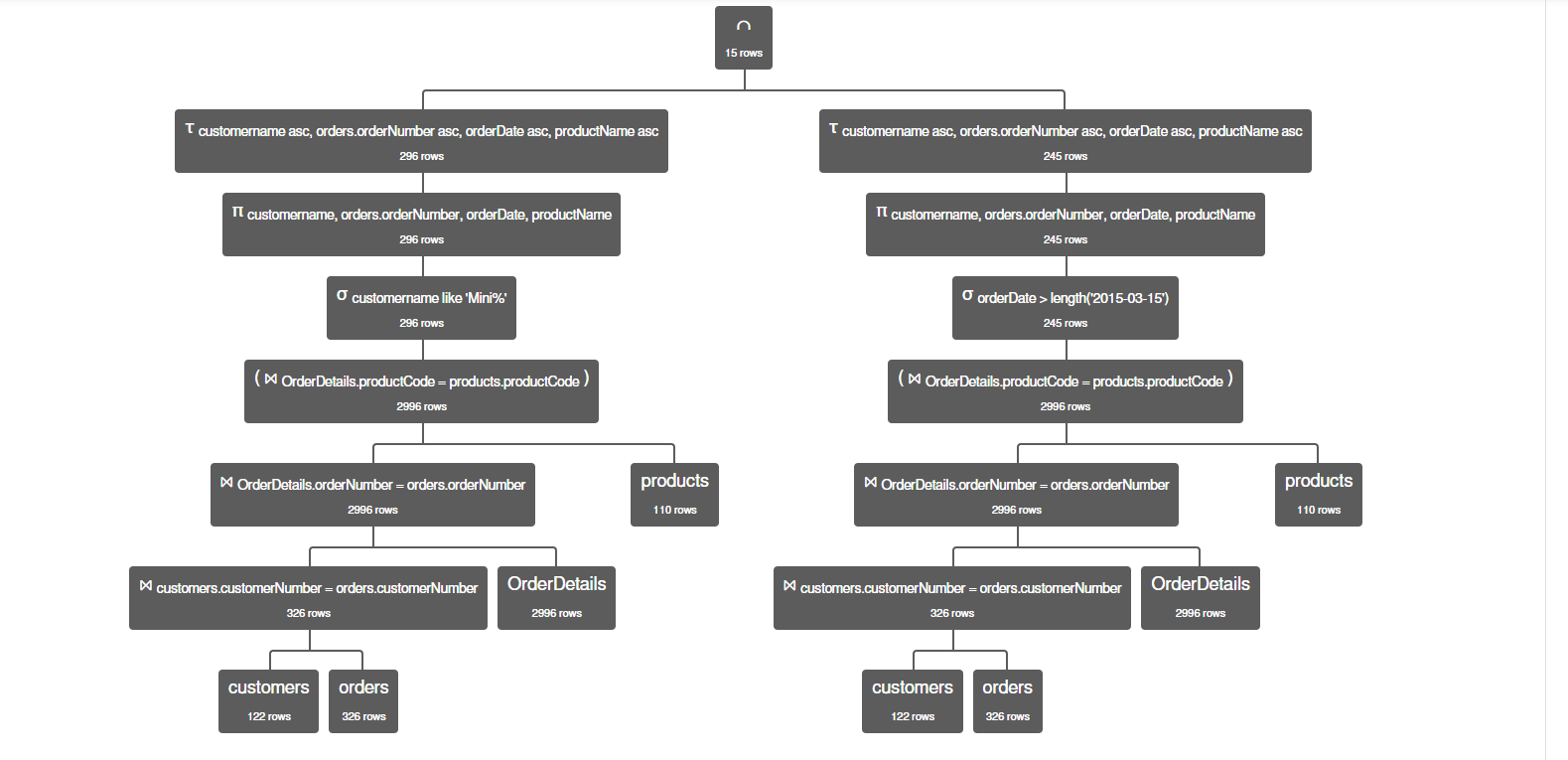
(customers ⨝ customers.customerNumber = orders.customerNumber

orders ⨝ OrderDetails.orderNumber = orders.orderNumber

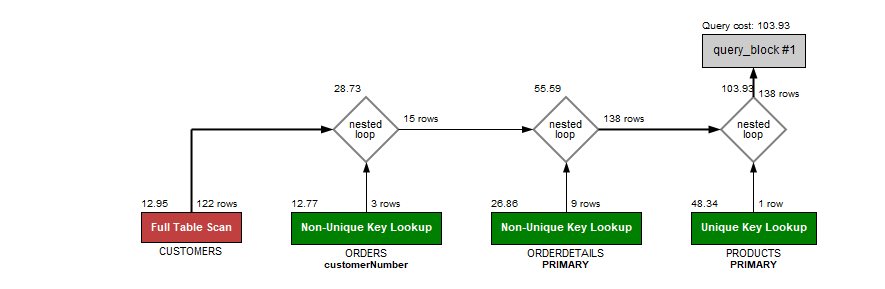
OrderDetails ⨝ OrderDetails.productCode = products.productCode

products)

**Image:**



**Execution (SQL visual explain):**



**Discussion:**

The execution explain is similar to part 1:Basic

**#3 Parenthesis**

SELECT CUSTOMERNAME, ORDERS.ORDERNUMBER, ORDERDATE, PRODUCTNAME FROM ORDERDETAILS

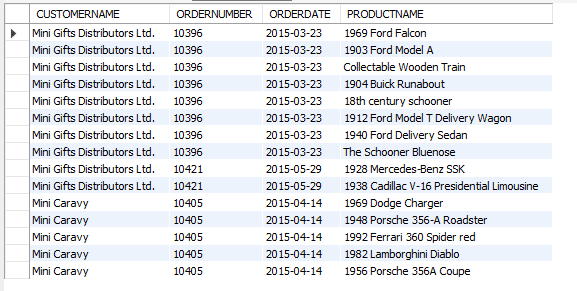
INNER JOIN PRODUCTS ON PRODUCTS.PRODUCTCODE=ORDERDETAILS.PRODUCTCODE

INNER JOIN ORDERS ON ORDERS.ORDERNUMBER=ORDERDETAILS.ORDERNUMBER

INNER JOIN CUSTOMERS ON CUSTOMERS.CUSTOMERNUMBER =ORDERS.CUSTOMERNUMBER

WHERE CUSTOMERNAME LIKE '%Mini%' AND ORDERDATE > '2015-03-15';

**Image:**

****

**Relational Algebra:**

πcustomername, orders.orderNumber, orderDate, productName

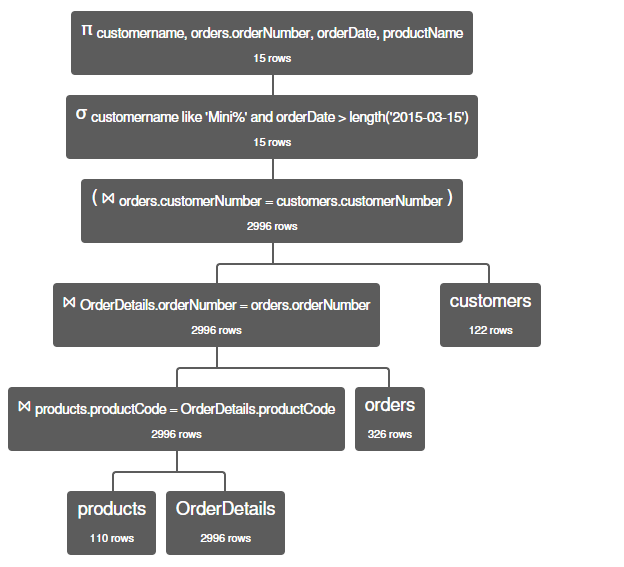
σcustomername like 'Mini%' ∧ orderDate > date('2015-03-15')

(products ⨝ products.productCode= OrderDetails.productCode

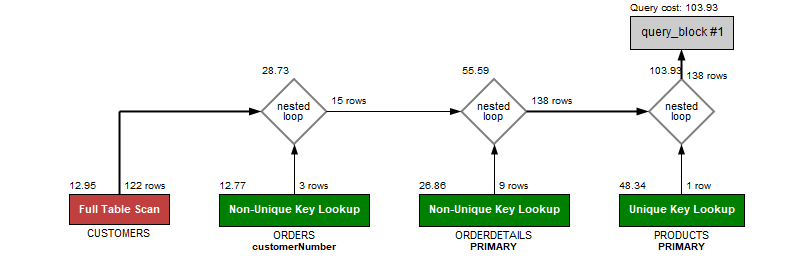
OrderDetails ⨝ OrderDetails.orderNumber=orders.orderNumber

orders⨝ orders.customerNumber=customers.customerNumber customers)

**Image:**



**Execution (SQL visual explain):**



**Discussion:**

We get a similar number of rows from both sql queries and relational algebra. We take the first relational algebra and change the join order, similar to what we did with the SQL queries. And the SQL visual explain still similar to the previous two

**#4 Date Digging**

**SQL Queries:**

SELECT CUSTOMERNAME, ORDERS.ORDERNUMBER, ORDERDATE, PRODUCTNAME FROM CUSTOMERS INNER JOIN ORDERS

USING (CUSTOMERNUMBER)

INNER JOIN ORDERDETAILS ON ORDERS.ORDERNUMBER=ORDERDETAILS.ORDERNUMBER

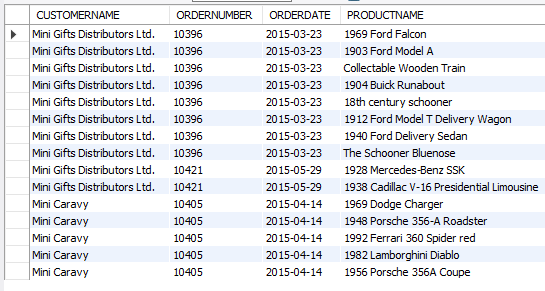
INNER JOIN PRODUCTS ON PRODUCTS.PRODUCTCODE=ORDERDETAILS.PRODUCTCODE

WHERE CUSTOMERNAME LIKE '%Mini%' AND (year(orderDate) > 2015 OR

(year(orderDate) = 2015 AND month(orderDate) > 3) or

(year(orderDate) = 2015 AND month(orderDate) = 3) and day(orderDate) > 15);

**Image:**



**Relational Algebra:**

πcustomername, orders.orderNumber, orderDate, productName

σcustomername like 'Mini%' ∧ (year(orderDate) > 2015 ∨

(year(orderDate) = 2015 ∧ month(orderDate) > 3) ∨

(year(orderDate) = 2015 ∧ month(orderDate) = 3) ∧

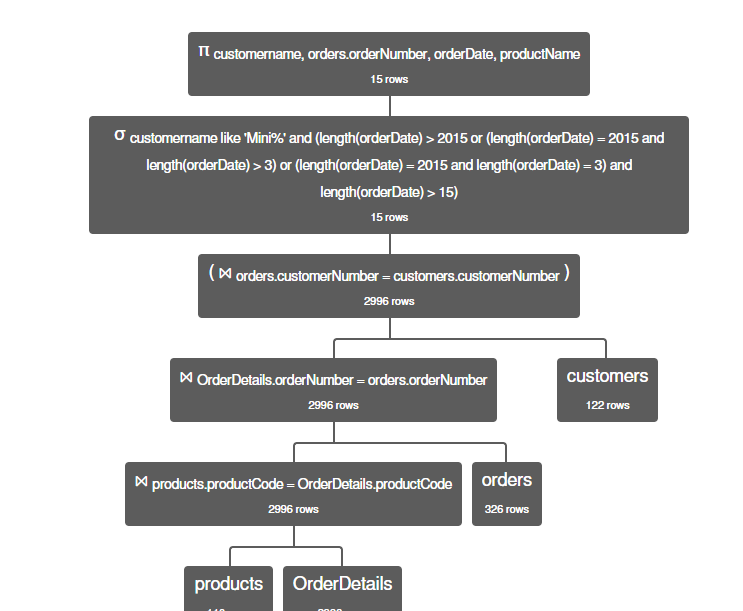
day(orderDate) > 15)

(products ⨝ products.productCode= OrderDetails.productCode

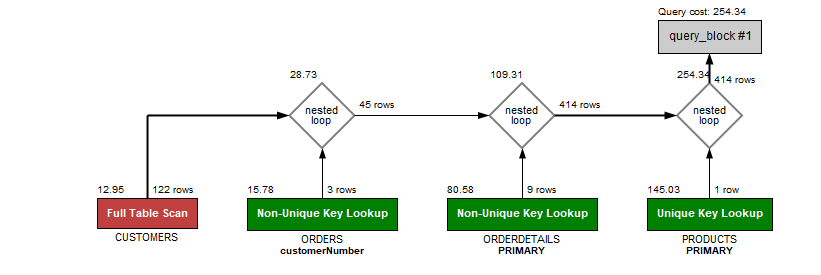
OrderDetails ⨝ OrderDetails.orderNumber=orders.orderNumber

orders⨝ orders.customerNumber=customers.customerNumber customers)

**Image:**



**Execution (SQL visual explain):**



**Discussion:**

We did not change much in order to get the relational algebra except the And and or key word to their symbols. The result is the same. Again, the SQL visual explain is the same as before but the run time is much worst.

**#5 Index**

**SQL Queries:**

CREATE INDEX Customer\_index ON CUSTOMERS(CUSTOMERNAME);

SELECT CUSTOMERNAME, ORDERS.ORDERNUMBER, ORDERDATE, PRODUCTNAME FROM CUSTOMERS INNER JOIN ORDERS

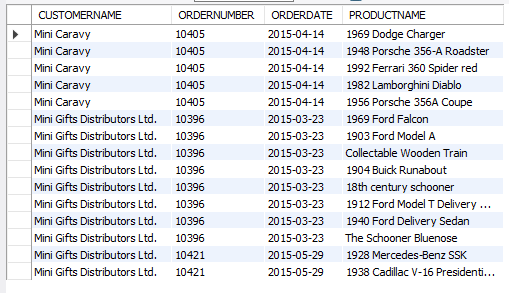
USING (CUSTOMERNUMBER)

INNER JOIN ORDERDETAILS ON ORDERS.ORDERNUMBER=ORDERDETAILS.ORDERNUMBER

INNER JOIN PRODUCTS ON PRODUCTS.PRODUCTCODE=ORDERDETAILS.PRODUCTCODE

WHERE CUSTOMERNAME LIKE '%Mini%' AND ORDERDATE > '2015-03-15';

Image:



**Relational Algebra:**

τcustomername, orders.orderNumber, orderDate, productName

πcustomername, orders.orderNumber, orderDate, productName

σcustomername like 'Mini%' ∧ orderDate > date('2015-03-15')

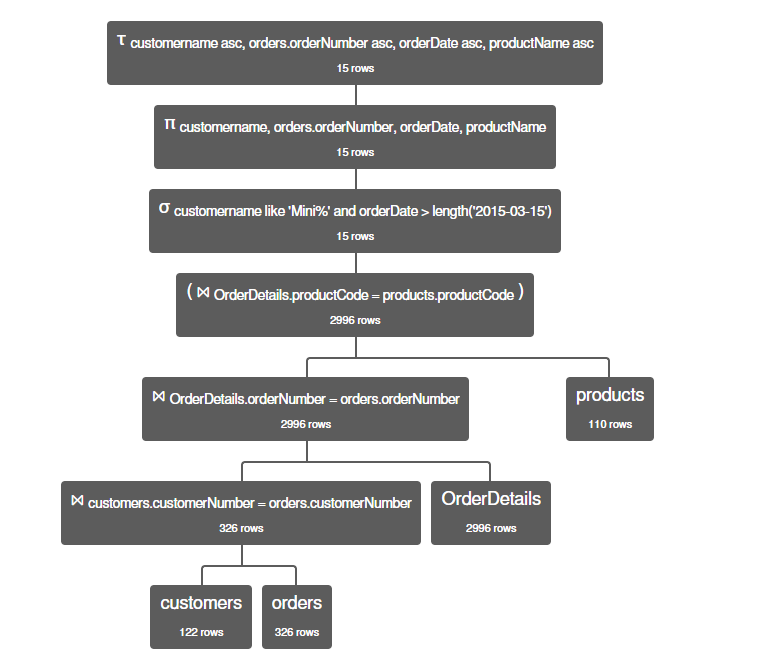
(customers ⨝ customers.customerNumber = orders.customerNumber

orders ⨝ OrderDetails.orderNumber = orders.orderNumber

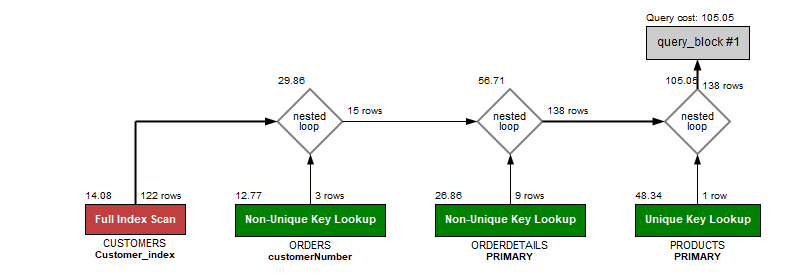
OrderDetails ⨝ OrderDetails.productCode = products.productCode

products)

**Image:**



**Execution (SQL visual explain):**



**Discussion:**

Similar result and execution, but the run time is much better than the previous one.