INFORME DESARROLLO SUPERANDES

Juan Sebastián Bravo, Juan Esteban Méndez

Universidad de los Andes, Bogotá, Colombia

{js.bravo, je.mendez[}@uniandes.edu.co](mailto:%7d@uniandes.edu.co)

Fecha de presentación: Diciembre 2 del 2018

Tabla de contenido

[Introducción 1](#_Toc531548926)

[Modelos solicitados 1](#_Toc531548927)

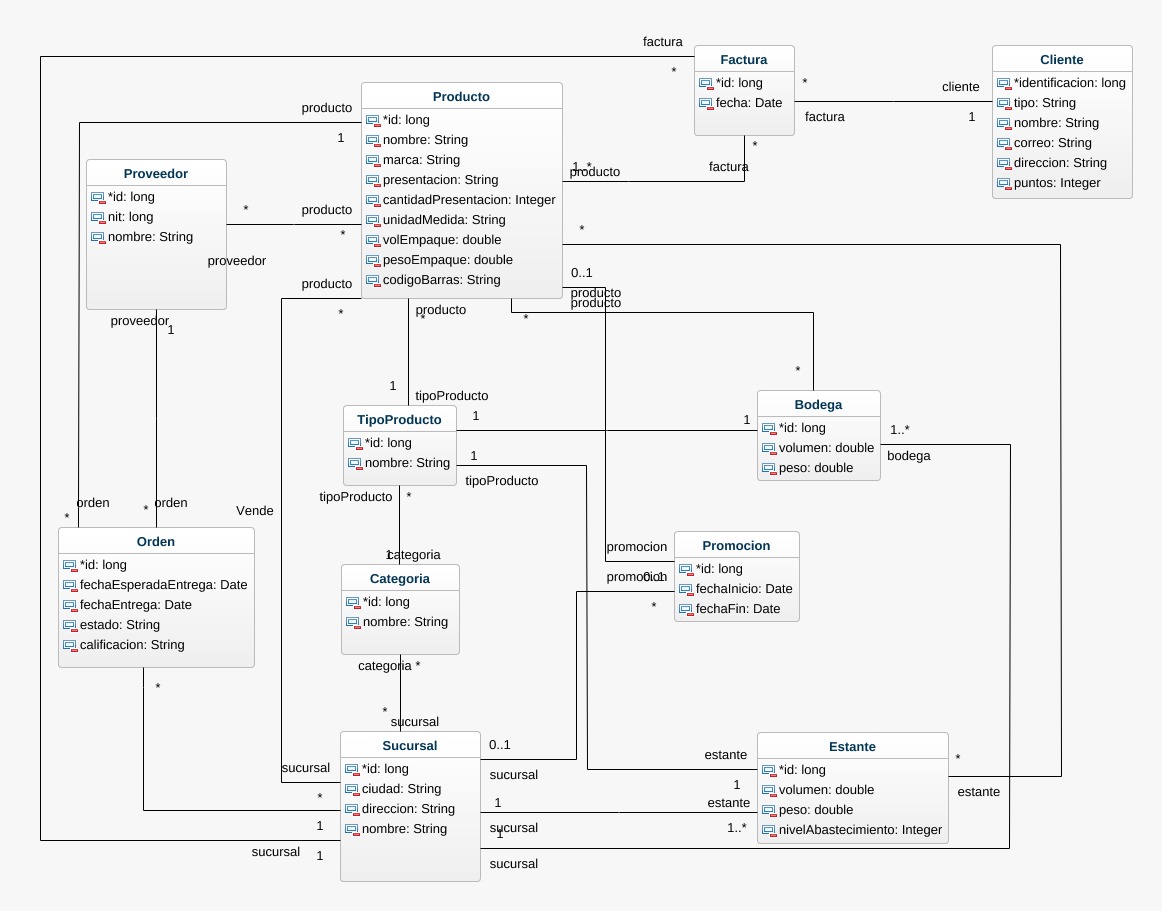
[Requerimientos funcionales de consulta iteración 3 3](#_Toc531548928)

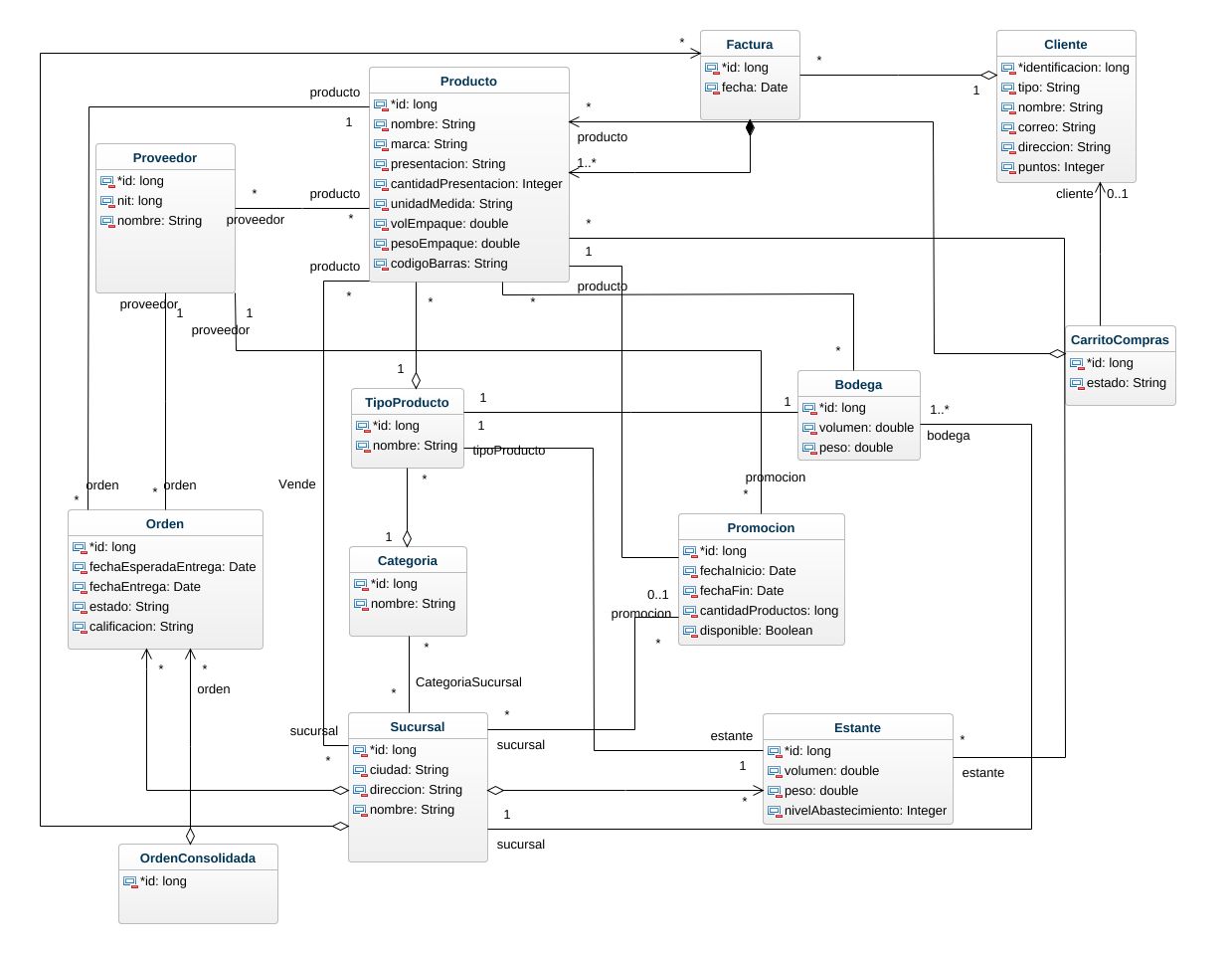
# Introducción

Este documento contiene la documentación sobre el caso de estudio de SuperAndes de la materia ISIS2304 – Sistemas Transaccionales tratado por el grupo B-05 de la materia en la tercera iteración del segundo semestre de 2018. En el siguiente documento se describe los modelos UML y relacionales necesarios y utilizados para el desarrollo de la aplicación que busca suplir los requerimientos de SuperAndes. Durante el documento se presentan los modelos en imágenes que permiten comprender la solución dada para los requerimientos funcionales de consulta propuestos, analizando su costo y el beneficio que trae el uso de índices en sistema manejador de bases de datos.

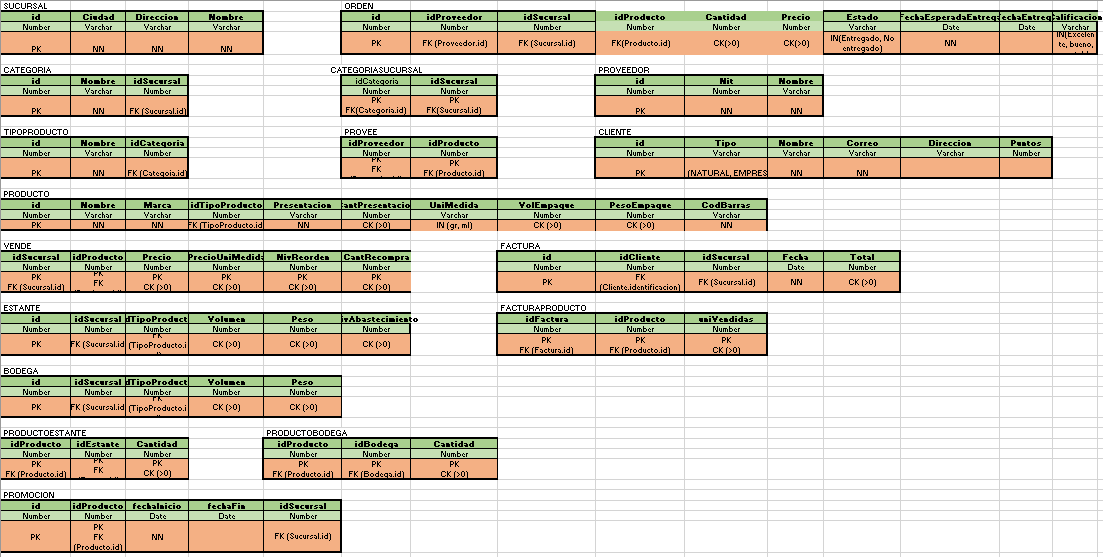
# Modelos solicitados

* 1. **UML (De la primera iteración)**

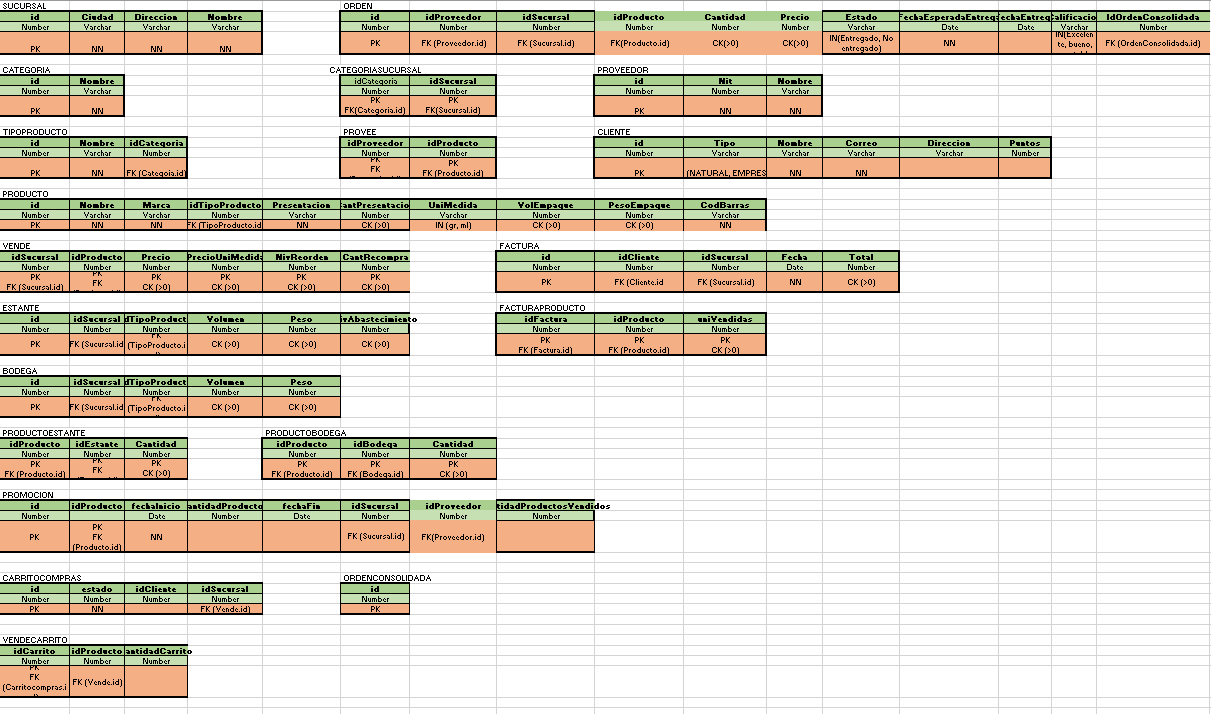


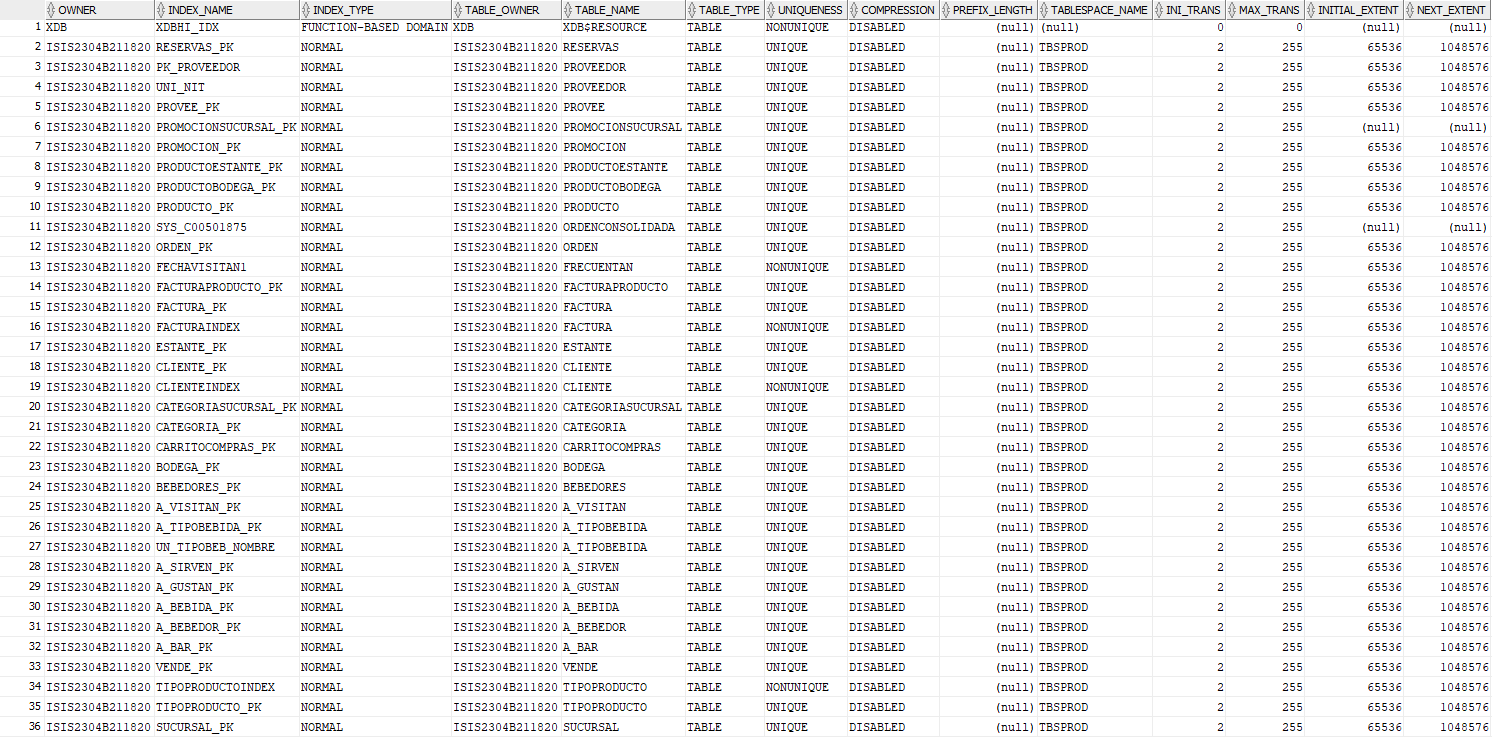
* 1. **UML actualizado**

1. **Modelo relacional (De la primera iteración)**



**Modelo relacional actualizado**



1. **Índices creados**

En la imagen se encuentran listados los índices que se encuentran en la base de datos de SuperAndes. En detalle, Oracle creó automáticamente todos los índices relacionados con las llaves primarias de las tablas. También se encuentran los índices CLIENTEINDEX y TIPOPRODUCTOINDEX, los cuales son explicados más adelante.

# Requerimientos funcionales de consulta iteración 3

* Consultar consumo en SuperAndes

SELECT CLIENTE.ID, CLIENTE.NOMBRE, FACTURA.ID, FACTURA.IDSUCURSAL,

FACTURA.FECHA, FACTURA.TOTAL, FACTURAPRODUCTO.IDPRODUCTO,

FACTURAPRODUCTO.UNIVENDIDAS

FROM CLIENTE

INNER JOIN FACTURA ON CLIENTE.ID = FACTURA.IDCLIENTE

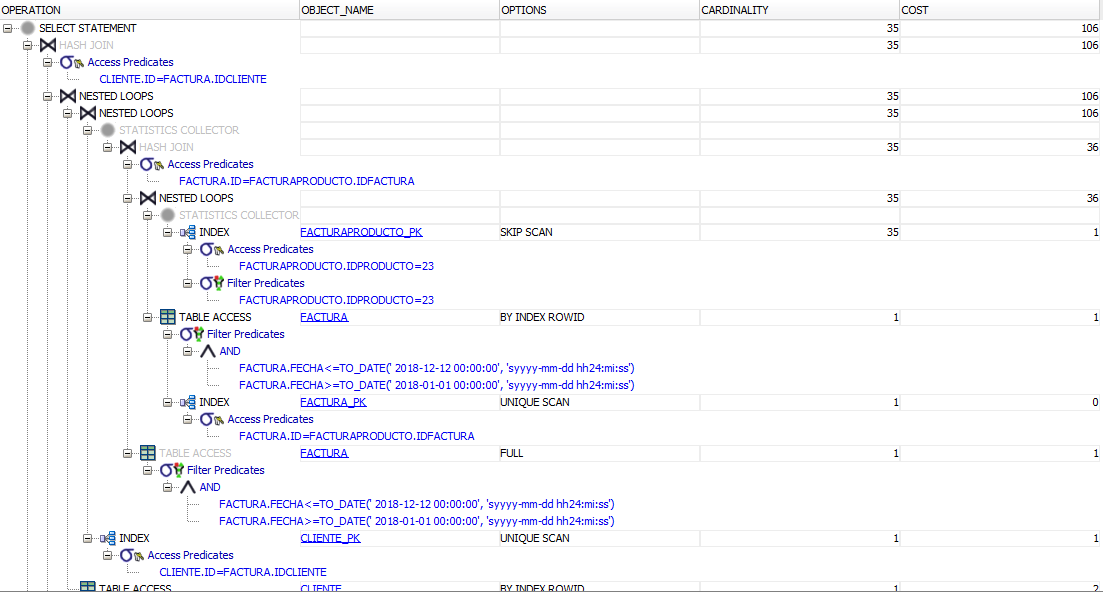
INNER JOIN FACTURAPRODUCTO ON FACTURA.ID = FACTURAPRODUCTO.IDFACTURA

WHERE FACTURAPRODUCTO.IDPRODUCTO = 23 AND FACTURA.FECHA BETWEEN '01/01/2018' AND '12/12/2018';

-- TABLAS ACCEDIDAS

-- CARDINALIDAD CLIENTE: 1’023.057   
-- CARDINALIDAD FACTURA: 270.163  
-- CARDINALIDAD FACTURAPRODUCTO: 257.559

-- TIEMPO DE RESPUESTA: 0.056 segundos.



Viendo la explicación del plan de Oracle, notamos que tanto el costo como la cardinalidad de la operación es mucho más bajo de lo que esperábamos, reduciéndose por el índice compuesto FACTURAPRODUCTO.PK, puesto que la búsqueda es por llave primaría. La creación de un índice en este requerimiento no es necesario, puesto que el índice automáticamente generado por Oracle FACTURAPRODUCTO.PK es el único necesario, utilizando todos los campos que posee FACTURAPRODUCTO.

*Cambiando los parámetros*

SELECT CLIENTE.ID, CLIENTE.NOMBRE, FACTURA.ID, FACTURA.IDSUCURSAL,

FACTURA.FECHA, FACTURA.TOTAL, FACTURAPRODUCTO.IDPRODUCTO,

FACTURAPRODUCTO.UNIVENDIDAS

FROM CLIENTE

INNER JOIN FACTURA ON CLIENTE.ID = FACTURA.IDCLIENTE

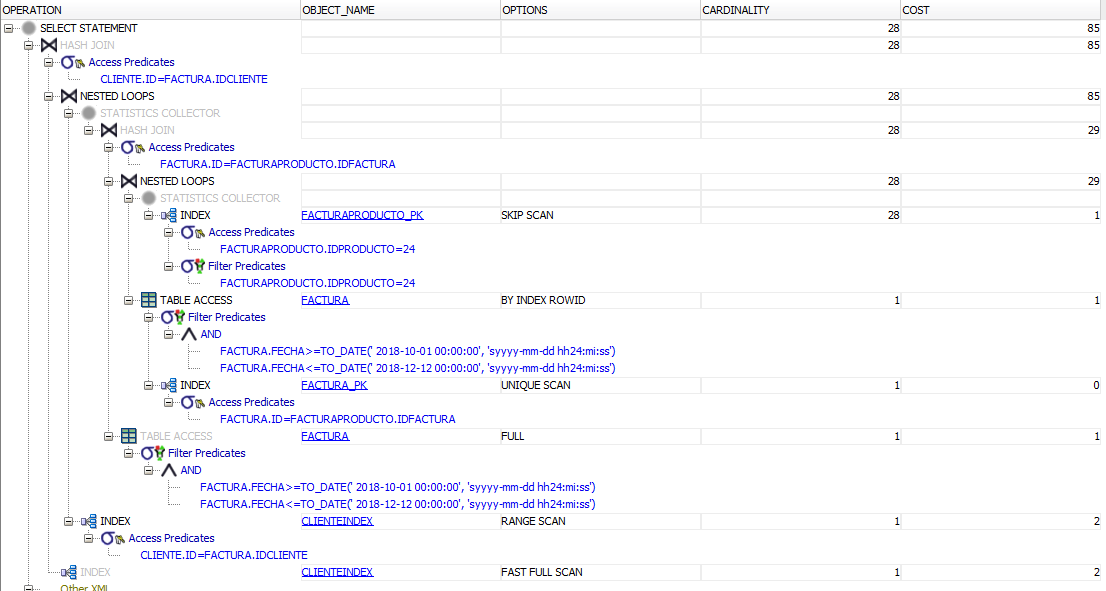
INNER JOIN FACTURAPRODUCTO ON FACTURA.ID = FACTURAPRODUCTO.IDFACTURA

WHERE FACTURAPRODUCTO.IDPRODUCTO = 24 AND FACTURA.FECHA BETWEEN '01/10/2018' AND '12/12/2018';

-- TABLAS ACCEDIDAS

-- CARDINALIDAD CLIENTE: 1’023.057   
-- CARDINALIDAD FACTURA: 270.163  
-- CARDINALIDAD FACTURAPRODUCTO: 257.559

-- TIEMPO DE RESPUESTA: 0.016 segundos.



Cambiando los parámetros de *fecha final* de la operación y el *id del producto,* el costo y la cardinalidad de la operación se redujeron de manera significativa, así como su tiempo de respuesta.

* Consultar NO consumo en SuperAndes

Notamos que este requerimiento funcional de consulta es el complemento de requerimiento funcional de consulta explicando en el literal anterior, por lo cual decidimos usar MINUS del conjunto total.

SELECT \* FROM (SELECT CLIENTE.ID, CLIENTE.NOMBRE, TRUNC(FACTURA.FECHA) AS

FECHA, SUM (FACTURAPRODUCTO.UNIVENDIDAS) AS SUMA\_TOT

FROM CLIENTE

FULL OUTER JOIN FACTURA ON CLIENTE.ID = FACTURA.IDCLIENTE

FULL OUTER JOIN FACTURAPRODUCTO ON FACTURA.ID = FACTURAPRODUCTO.IDFACTURA

GROUP BY CLIENTE.ID, CLIENTE.NOMBRE, TRUNC(FACTURA.FECHA)

MINUS (SELECT CLIENTE.ID, CLIENTE.NOMBRE, TRUNC(FACTURA.FECHA), SUM (FACTURAPRODUCTO.UNIVENDIDAS) AS SUMA\_TOT

FROM CLIENTE

FULL OUTER JOIN FACTURA ON CLIENTE.ID = FACTURA.IDCLIENTE

FULL OUTER JOIN FACTURAPRODUCTO ON FACTURA.ID = FACTURAPRODUCTO.IDFACTURA

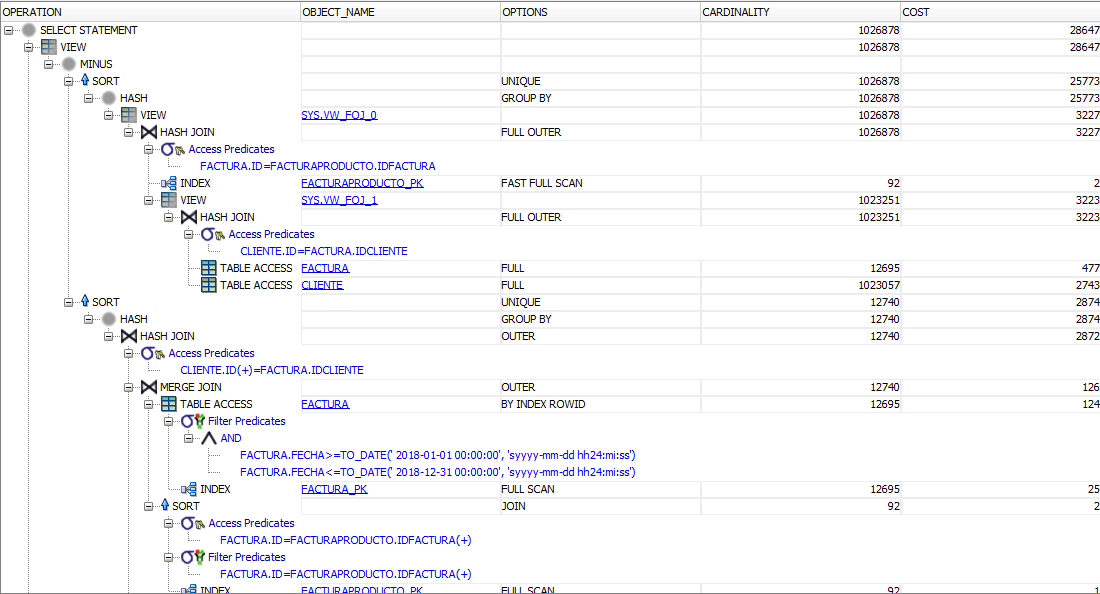
WHERE FACTURA.FECHA BETWEEN '01/01/2018' AND '31/12/2018'

GROUP BY CLIENTE.ID, CLIENTE.NOMBRE, TRUNC(FACTURA.FECHA)))

-- TABLAS ACCEDIDAS

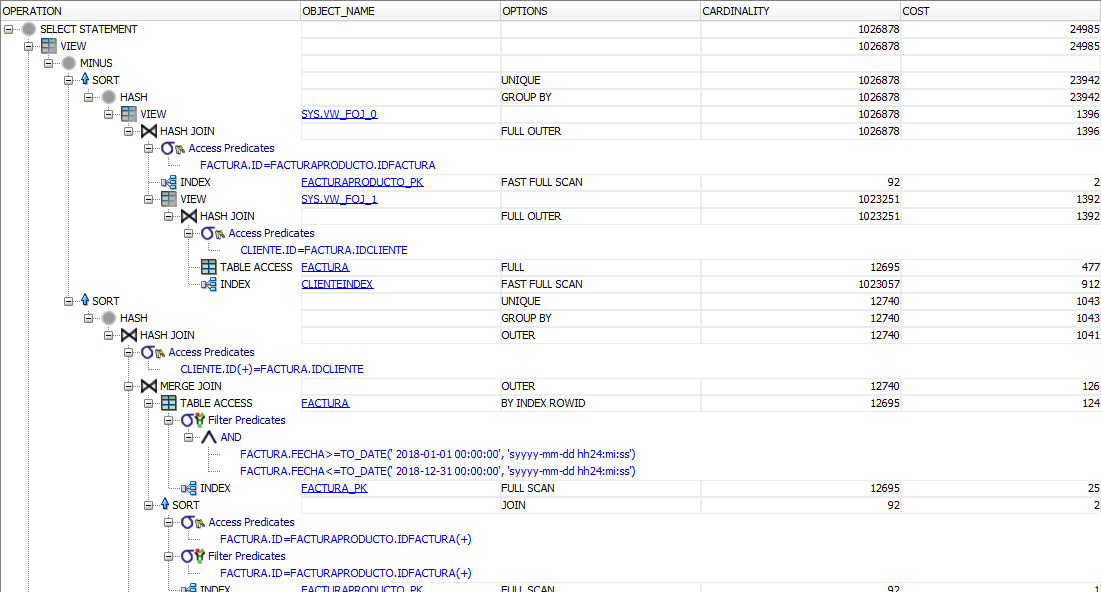
-- CARDINALIDAD CLIENTE: 1’023.057   
-- CARDINALIDAD FACTURA: 270.163  
-- CARDINALIDAD FACTURAPRODUCTO: 257.559

-- TIEMPO DE RESPUESTA: 5 segundos.



Analizando la operación de consulta, decidimos crear un índice compuesto denominado CLIENTEINDEX en los campos ID y NOMBRE de CLIENTE. Lo anterior debido a que la consulta utiliza continuamente estos campos de la tabla CLIENTE, por lo cual un índice podría filtrar más rápidamente.

CREATE INDEX CLIENTEINDEX ON CLIENTE (ID, NOMBRE)



Revisando de nuevo el plan de consulta después de haber creado el índice compuesto CLIENTEINDEX, vemos que el costo disminuyó notablemente.

NUEVO TIEMPO DE RESPUESTA: 3.8 segundos

*Cambiando los parámetros*

SELECT \* FROM (SELECT CLIENTE.ID, CLIENTE.NOMBRE, TRUNC(FACTURA.FECHA) AS

FECHA

FROM CLIENTE

FULL OUTER JOIN FACTURA ON CLIENTE.ID = FACTURA.IDCLIENTE

FULL OUTER JOIN FACTURAPRODUCTO ON FACTURA.ID = FACTURAPRODUCTO.IDFACTURA

GROUP BY CLIENTE.ID, CLIENTE.NOMBRE, TRUNC(FACTURA.FECHA)

MINUS (SELECT CLIENTE.ID, CLIENTE.NOMBRE, TRUNC(FACTURA.FECHA), SUM (FACTURAPRODUCTO.UNIVENDIDAS) AS SUMA\_TOT

FROM CLIENTE

FULL OUTER JOIN FACTURA ON CLIENTE.ID = FACTURA.IDCLIENTE

FULL OUTER JOIN FACTURAPRODUCTO ON FACTURA.ID = FACTURAPRODUCTO.IDFACTURA

WHERE FACTURA.FECHA BETWEEN '01/10/2018' AND '31/12/2018'

GROUP BY CLIENTE.ID, CLIENTE.NOMBRE, TRUNC(FACTURA.FECHA)))

-- TABLAS ACCEDIDAS

-- CARDINALIDAD CLIENTE: 1’023.057   
-- CARDINALIDAD FACTURA: 270.163  
-- CARDINALIDAD FACTURAPRODUCTO: 257.559

-- TIEMPO DE RESPUESTA: 4.8 segundos.

Cambiando los parámetros de *fecha final* de la operación y el *id de la sucursal* en la vista gerente*,* el costo y la cardinalidad de la operación se redujeron de manera significativa, así como su tiempo de respuesta.



* Consultar funcionamiento en SuperAndes

Notamos que este requerimiento funcional de consulta puede facilitarse con el uso de vistas, por lo cual creamos las siguientes:

CREATE VIEW PRODUCTOSVENDIDOSXSEMANA AS(

SELECT TO\_NUMBER(TO\_CHAR(TO\_DATE(FACTURA.FECHA,'DD/MM/YYYY'),'WW')) AS SEMANA, FACTURAPRODUCTO.IDPRODUCTO, SUM(FACTURAPRODUCTO.UNIVENDIDAS) AS SUMA

FROM FACTURA

INNER JOIN FACTURAPRODUCTO ON FACTURA.ID = FACTURAPRODUCTO.IDFACTURA

GROUP BY TO\_NUMBER(to\_char(to\_date(FACTURA.FECHA,'DD/MM/YYYY'),'WW')),FACTURAPRODUCTO.IDPRODUCTO

);

CREATE VIEW PEDIDOSPROVEEDORXSEMANA AS(

SELECT TO\_NUMBER(to\_char(to\_date(ORDEN.FECHAESPERADAENTREGA,'DD/MM/YYYY'),'WW')) AS SEMANA, IDPROVEEDOR, COUNT(IDPROVEEDOR) AS NUM\_PEDIDOS

FROM ORDEN

GROUP BY TO\_NUMBER(to\_char(to\_date(ORDEN.FECHAESPERADAENTREGA,'DD/MM/YYYY'),'WW')), IDPROVEEDOR);

* Caso 1: Producto más vendido en cada semana del año

SELECT PRODUCTOSVENDIDOSXSEMANA.SEMANA, PRODUCTOSVENDIDOSXSEMANA.IDPRODUCTO,

PRODUCTO.NOMBRE, SUBQUERY.MAXIMO

FROM

(SELECT SEMANA, MAX(SUMA) AS MAXIMO

FROM PRODUCTOSVENDIDOSXSEMANA

GROUP BY SEMANA) SUBQUERY

INNER JOIN PRODUCTOSVENDIDOSXSEMANA ON PRODUCTOSVENDIDOSXSEMANA.SUMA = SUBQUERY.MAXIMO AND PRODUCTOSVENDIDOSXSEMANA.SEMANA = SUBQUERY.SEMANA

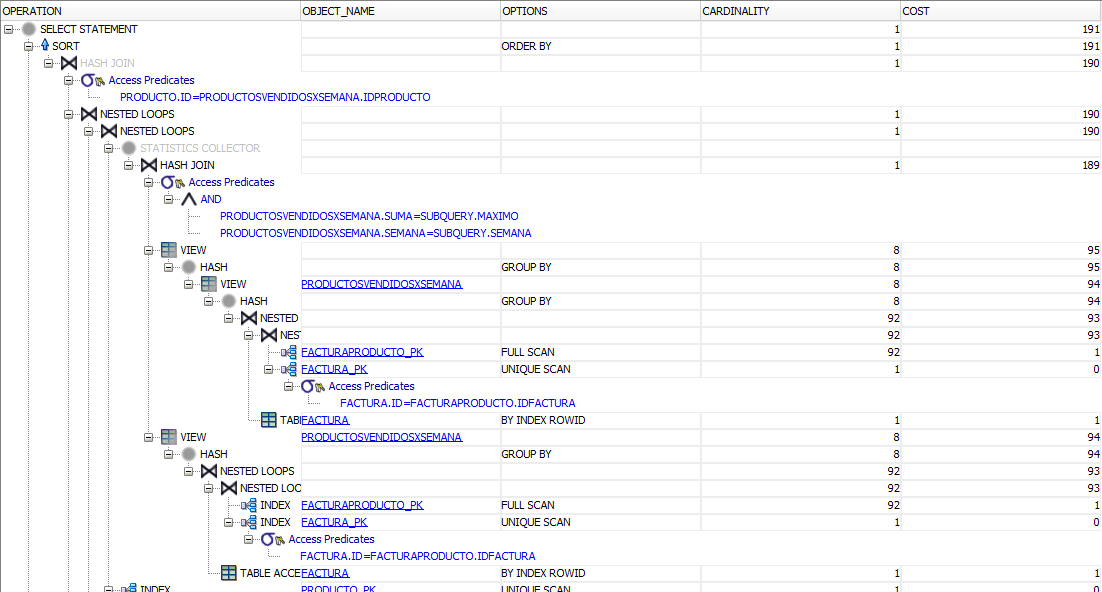
INNER JOIN PRODUCTO ON PRODUCTO.ID = PRODUCTOSVENDIDOSXSEMANA.IDPRODUCTO

ORDER BY PRODUCTOSVENDIDOSXSEMANA.SEMANA;

-- TABLAS ACCEDIDAS

-- CARDINALIDAD PRODUCTO: 2.563  
-- CARDINALIDAD FACTURA: 270.163  
-- CARDINALIDAD FACTURAPRODUCTO: 257.559

-- TIEMPO DE RESPUESTA: 3.1 segundos.



Notamos que con la vista creada llamada PRODUCTOSVENDIDOSXSEMANA se redujo el tiempo de respuesta de gran manera.

* Caso 2: Producto menos vendido en cada semana del año

SELECT PRODUCTOSVENDIDOSXSEMANA.SEMANA, PRODUCTOSVENDIDOSXSEMANA.IDPRODUCTO,

PRODUCTO.NOMBRE, SUBQUERY.MINIMO

FROM

(SELECT SEMANA, MIN(SUMA) AS MINIMO

FROM PRODUCTOSVENDIDOSXSEMANA

GROUP BY SEMANA) SUBQUERY

INNER JOIN PRODUCTOSVENDIDOSXSEMANA ON PRODUCTOSVENDIDOSXSEMANA.SUMA = SUBQUERY.MINIMO AND PRODUCTOSVENDIDOSXSEMANA.SEMANA = SUBQUERY.SEMANA

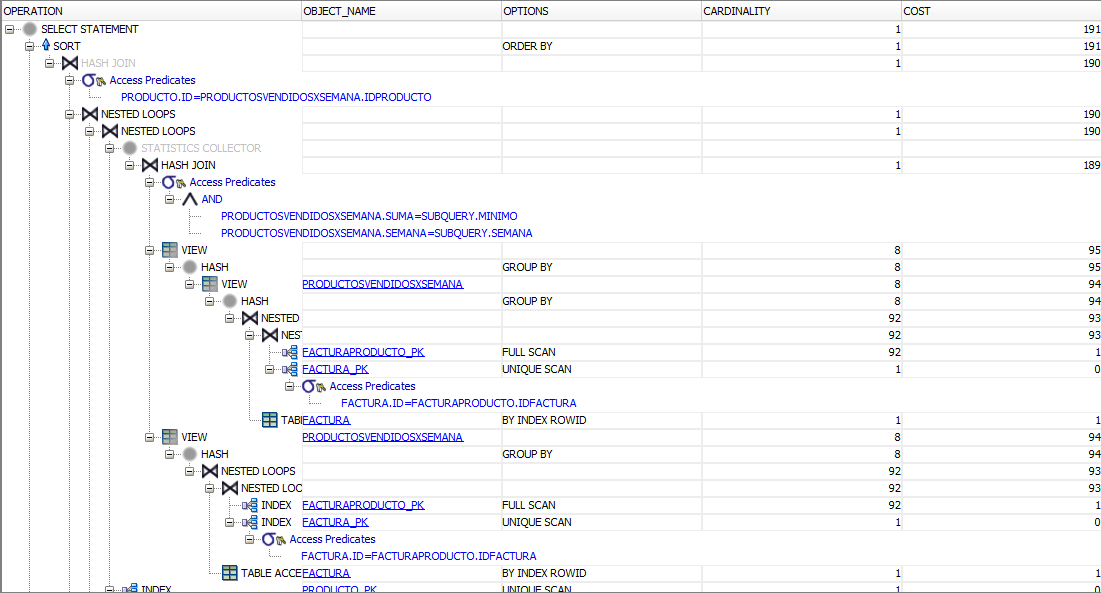
INNER JOIN PRODUCTO ON PRODUCTO.ID = PRODUCTOSVENDIDOSXSEMANA.IDPRODUCTO

ORDER BY PRODUCTOSVENDIDOSXSEMANA.SEMANA;

-- TABLAS ACCEDIDAS

-- CARDINALIDAD PRODUCTO: 2.563  
-- CARDINALIDAD FACTURA: 270.163  
-- CARDINALIDAD FACTURAPRODUCTO: 257.559

-- TIEMPO DE RESPUESTA: 1.5 segundos.



Notamos que con la vista creada llamada PRODUCTOSVENDIDOSXSEMANA se redujo el tiempo de respuesta de gran manera.

* Caso 3: Proveedor más solicitado en cada semana del año

SELECT PEDIDOSPROVEEDORXSEMANA.SEMANA, PEDIDOSPROVEEDORXSEMANA.IDPROVEEDOR ,

PROVEEDOR.NOMBRE, PEDIDOSPROVEEDORXSEMANA.NUM\_PEDIDOS

FROM

(SELECT SEMANA, MAX(NUM\_PEDIDOS) AS NUM\_PEDIDOS

FROM PEDIDOSPROVEEDORXSEMANA

GROUP BY SEMANA, IDPROVEEDOR) SUBQUERY

INNER JOIN PEDIDOSPROVEEDORXSEMANA ON PEDIDOSPROVEEDORXSEMANA.SEMANA = SUBQUERY.SEMANA AND PEDIDOSPROVEEDORXSEMANA.NUM\_PEDIDOS = SUBQUERY.NUM\_PEDIDOS

INNER JOIN PROVEEDOR ON PROVEEDOR.ID = PEDIDOSPROVEEDORXSEMANA.IDPROVEEDOR;

-- TABLAS ACCEDIDAS

-- CARDINALIDAD PROVEEDOR: 1.005  
-- CARDINALIDAD PROVEE: 1  
-- CARDINALIDAD ORDEN: 11

-- TIEMPO DE RESPUESTA: 0,032 segundos.



Notamos que con la vista creada llamada PEDIDOSPROVEEDORXSEMANA se redujo el tiempo de respuesta de gran manera.

* Caso 4: Proveedor menos solicitado en cada semana del año

SELECT PEDIDOSPROVEEDORXSEMANA.SEMANA, PEDIDOSPROVEEDORXSEMANA.IDPROVEEDOR ,

PROVEEDOR.NOMBRE, PEDIDOSPROVEEDORXSEMANA.NUM\_PEDIDOS

FROM

(SELECT SEMANA, MIN(NUM\_PEDIDOS) AS NUM\_PEDIDOS

FROM PEDIDOSPROVEEDORXSEMANA

GROUP BY SEMANA, IDPROVEEDOR) SUBQUERY

INNER JOIN PEDIDOSPROVEEDORXSEMANA ON PEDIDOSPROVEEDORXSEMANA.SEMANA = SUBQUERY.SEMANA AND PEDIDOSPROVEEDORXSEMANA.NUM\_PEDIDOS = SUBQUERY.NUM\_PEDIDOS

INNER JOIN PROVEEDOR ON PROVEEDOR.ID = PEDIDOSPROVEEDORXSEMANA.IDPROVEEDOR;

-- TABLAS ACCEDIDAS

-- CARDINALIDAD PROVEEDOR: 1.005  
-- CARDINALIDAD PROVEE: 1  
-- CARDINALIDAD ORDEN: 11

-- TIEMPO DE RESPUESTA: 0,032 segundos.



Notamos que con la vista creada llamada PEDIDOSPROVEEDORXSEMANA se redujo el tiempo de respuesta de gran manera.

* Consultar los buenos clientes
  + Caso 1: Clientes que hicieron al menos una compra por mes

SELECT CLIENTE.ID, CLIENTE.NOMBRE, EXTRACT(MONTH FROM FACTURA.FECHA) AS

MES, COUNT(DISTINCT FACTURA.ID) AS COMPRAS

FROM CLIENTE

INNER JOIN FACTURA ON CLIENTE.ID = FACTURA.IDCLIENTE

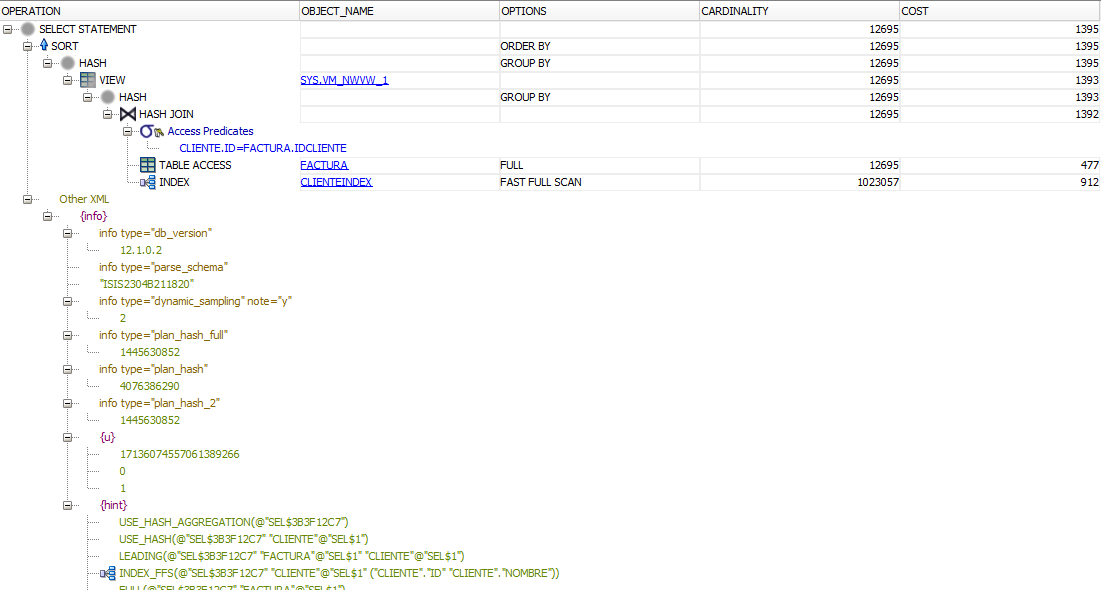
GROUP BY CLIENTE.ID, CLIENTE.NOMBRE,EXTRACT(MONTH FROM FACTURA.FECHA)

ORDER BY COUNT(DISTINCT FACTURA.ID) DESC,CLIENTE.NOMBRE ASC;

-- TABLAS ACCEDIDAS

-- CARDINALIDAD CLIENTE: 1’023.057   
-- CARDINALIDAD FACTURA: 270.163

-- TIEMPO DE RESPUESTA: 2 segundos.



Revisando el plan de Oracle, notamos que la creación del índice CLIENTEINDEX redujo el costo de la operación y el tiempo de respuesta.

NUEVO TIEMPO DE RESPUESTA: 1,7 segundos

* + Caso 2: Clientes que hicieron al menos una compra por mes

SELECT DISTINCT CLIENTE.ID, CLIENTE.NOMBRE

FROM CLIENTE

INNER JOIN FACTURA ON CLIENTE.ID = FACTURA.IDCLIENTE

INNER JOIN FACTURAPRODUCTO ON FACTURA.ID = FACTURAPRODUCTO.IDFACTURA

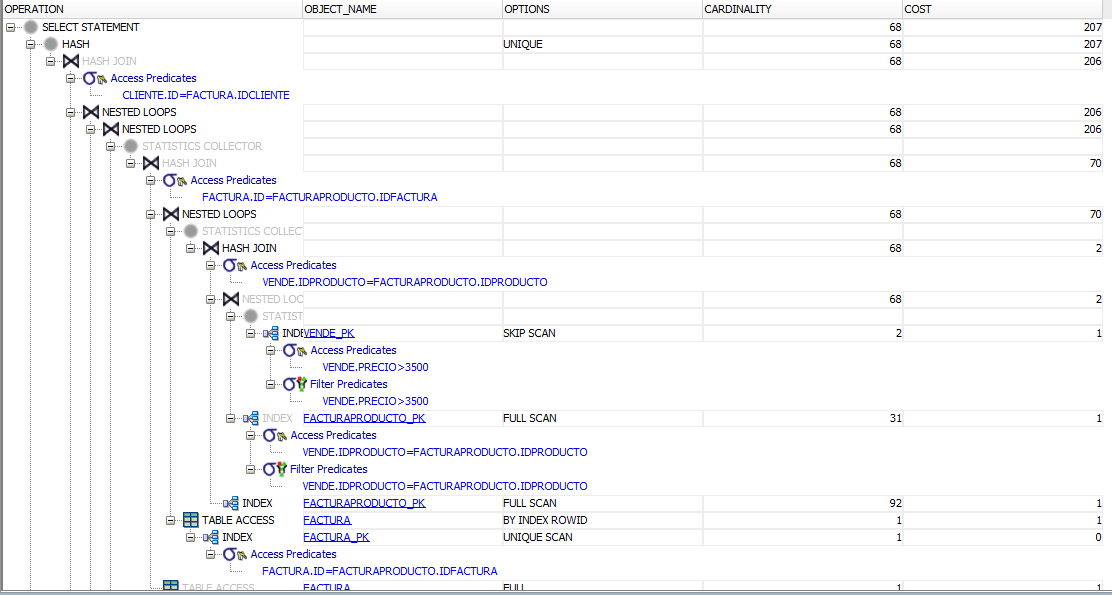
INNER JOIN VENDE ON VENDE.IDPRODUCTO = FACTURAPRODUCTO.IDPRODUCTO

WHERE VENDE.PRECIO > 100000;

-- TABLAS ACCEDIDAS

-- CARDINALIDAD CLIENTE: 1’023.057   
-- CARDINALIDAD FACTURA: 270.163  
-- CARDINALIDAD FACTURAPRODUCTO: 257.559

-- TIEMPO DE RESPUESTA: 0,064 segundos.



* + Caso 3: Clientes que han realizado una compra de un producto de ‘Electronics’ y ‘Tools’

SELECT DISTINCT CLIENTE.ID, CLIENTE.NOMBRE

FROM CLIENTE

INNER JOIN FACTURA ON CLIENTE.ID = FACTURA.IDCLIENTE

INNER JOIN FACTURAPRODUCTO ON FACTURA.ID = FACTURAPRODUCTO.IDFACTURA

INNER JOIN PRODUCTO ON PRODUCTO.ID = FACTURAPRODUCTO.IDPRODUCTO

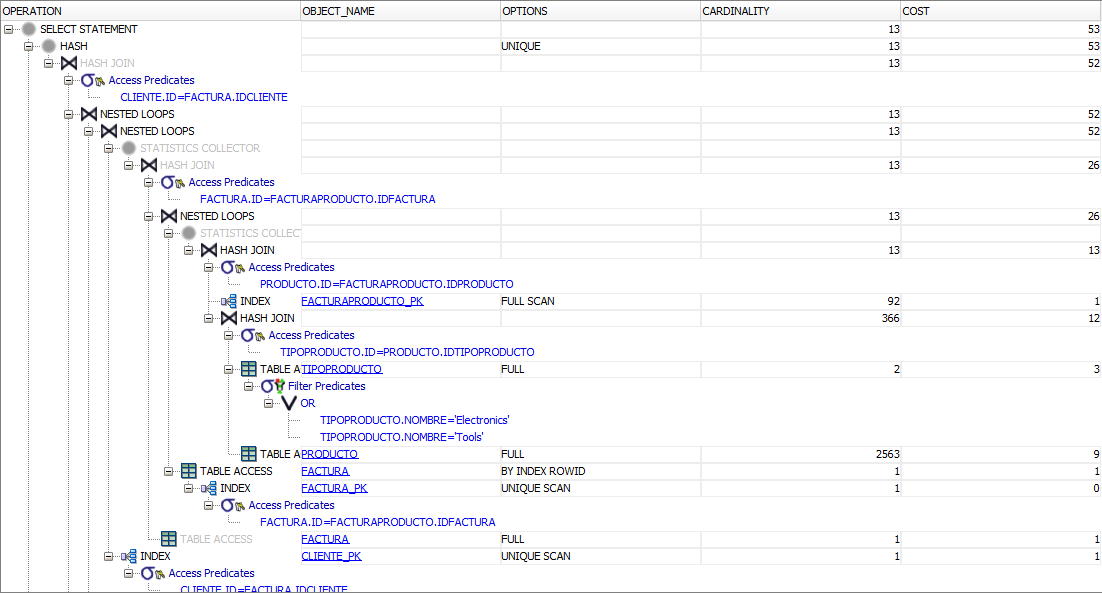
INNER JOIN TIPOPRODUCTO ON TIPOPRODUCTO.ID = PRODUCTO.IDTIPOPRODUCTO

WHERE TIPOPRODUCTO.NOMBRE = 'Electronics' or TIPOPRODUCTO.NOMBRE = 'Tools';

-- CARDINALIDAD CLIENTE: 1’023.057

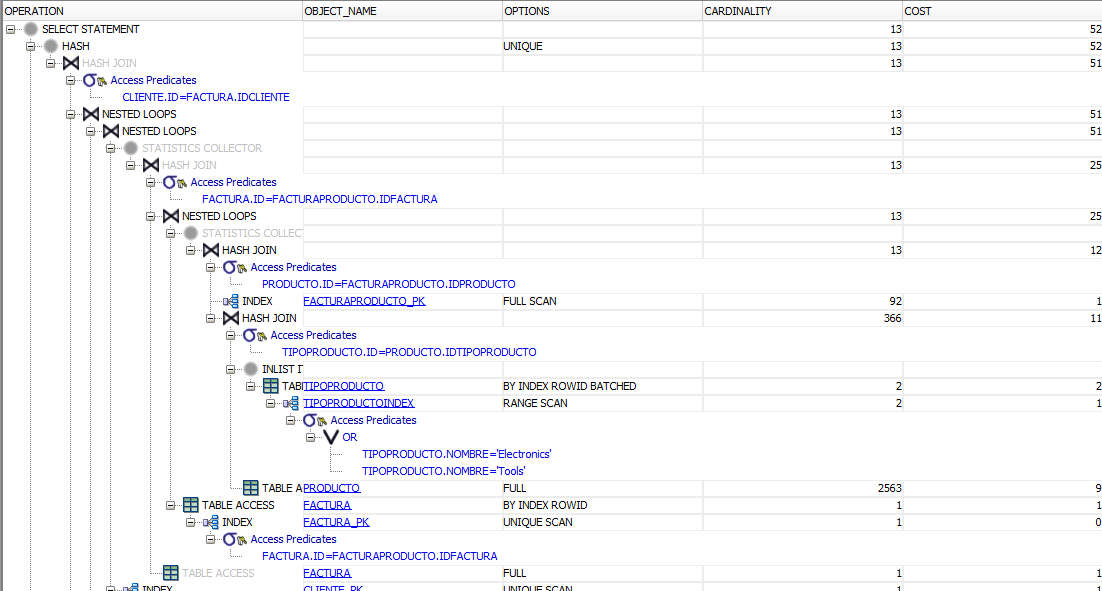
-- CARDINALIDAD PRODUCTO: 2.563  
-- CARDINALIDAD FACTURA: 270.163  
-- CARDINALIDAD FACTURAPRODUCTO: 257.559

-- TIEMPO DE RESPUESTA: 1 segundo.



Analizando la operación de consulta, decidimos crear un índice simple denominado TIPOPRODUCTOINDEX en el campo NOMBRE de TIPOPRODUCTO. Lo anterior debido a que la consulta filtra este campo, por lo cual un índice podría filtrar más rápidamente.

CREATE INDEX CLIENTEINDEX ON CLIENTE (ID, NOMBRE)



Revisando de nuevo el plan de consulta después de haber creado el índice simple TIPOPRODUCTOINDEX, vemos que el costo disminuyó.

NUEVO TIEMPO DE RESPUESTA: 0,044 segundos