**Diseño de la aplicación**

**Cambios respecto a la anterior iteración:**

No se incorporaron nuevos cambios a la aplicación, se intentó hacer la menor cantidad de cambios posibles debido a la complejidad de la iteración; esto, con el objetivo de reducir el tiempo en el desarrollo de la aplicación. Se dio un enfoque a crear datos semi-reales y con la suficiente variedad y tamaño para tener resultados variados entre consultas. También, se le dio énfasis a generar la suficiente cantidad de datos en la población de las tablas.

**Diseño físico de la aplicación:**

**Índices y requerimientos funcionales:** Debido a la gran cantidad de recursos generados, es necesario optimizar la base de datos para poder responder más rápidamente. De aquí nace el uso de los índices para optimizar las consultas establecidas. Como cada tabla requiere de diferentes tablas y campos, se explicarán los índices utilizados sobre cada consulta. El código SQL de cada sentencia se encuentra en SQL/RFC Iteración 4.sql para no saturar el documento.

* **RFC 9:**

En este requerimiento era necesario conocer los datos de los usuarios a partir de ciertos patrones en las compras, en particular el restaurante donde se compra en determinado rango de fechas. Para

Esta consulta hace uso de los campos en las tablas CHECKOUT y PRODUCTO\_CHECKOUT, sobre los campos TIEMPOR (fecha), ID\_CHECKOUT y USUARIO\_CLIENTE por lo que se hace necesario implementar índices sobre esas tablas y esos campos. Se decidió utilizar un índice de XXX puesto que ZZZ.

* **RFC 10:**

De manera similar al anterior, se requería consultar el complemento del conjunto que representa la consulta anterior, por lo que los campos involucrados son los mismos. Este requerimiento no era una modificación trivial del otro, puesto que si se usa IS NOT NULL en algún lugar de la consulta, se anulaba la utilidad de cualquier índice.

Los índices que actúan sobre esta consulta son 111, 222, 333, ya que al igual que el anterior requerimiento, ayudan a disminuir los accesos a disco de encontrar los campos a comparar en el Join.

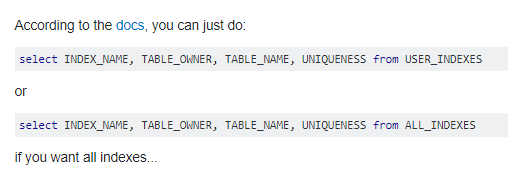
* **RFC 11:**

Para este requerimiento era necesario efectuar varias funciones agregadas sobre las tablas encargadas de modelar las ventas, CHECKOUT y PRODUCTO\_CHECKOUT. Cómo los árboles B+ manejan mejor las relaciones de orden y estamos buscando máximos / mínimos, es ideal utilizarlos en esta situación sobre los campos donde se ejecutan las funciones de orden (min, max). Por otra parte, para las funciones de conteo, un índice de Bitmap es mejor puesto que la cardinalidad de la categorización que usamos para las fechas es 1/7 (7 días de la semana), lo cual es bastante útil al categorizar más de 1 millón de compras.

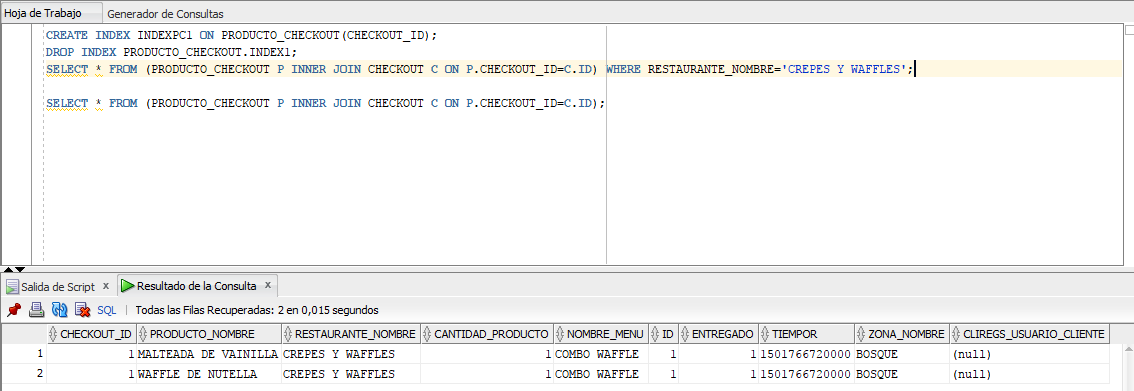
* **RFC 12:**

El último requerimiento nos solicitaba extraer la información de aquellos clientes que cumplan varias condiciones, por lo que es necesario hacer un JOIN de las diferentes condiciones para ser ‘un buen cliente’.

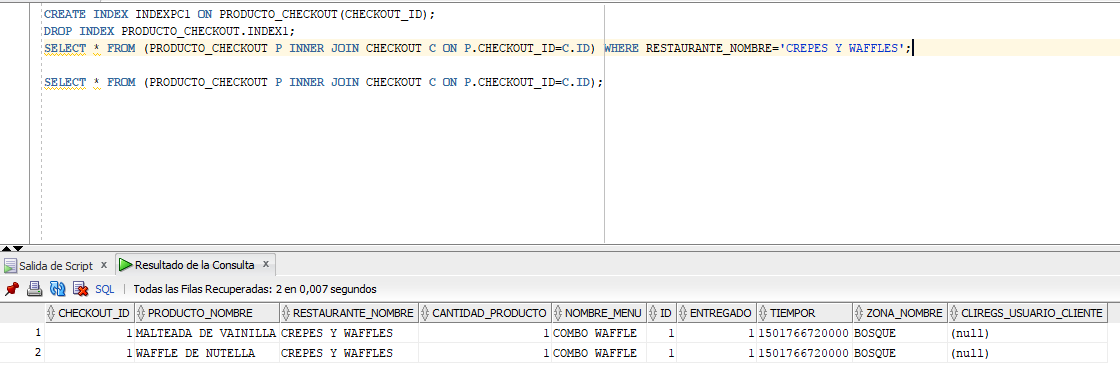
**Índices creados automáticamente por Oracle:**

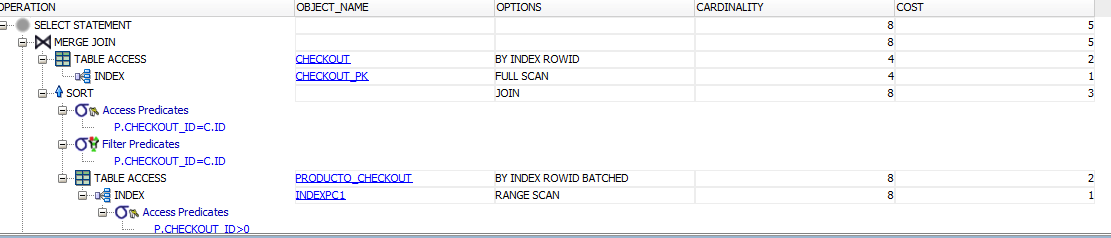
*  [insertar foto]

JOIN 1 consulta 1



No index, 0.015 segundos

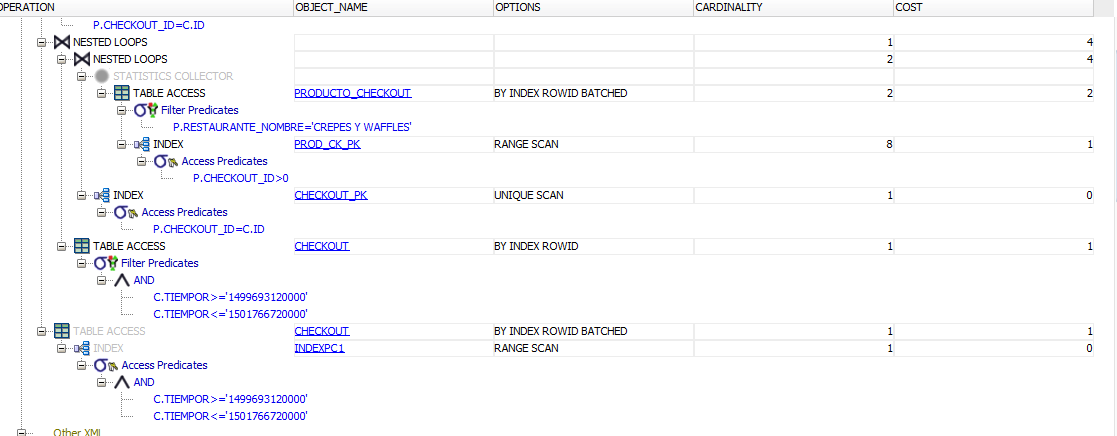




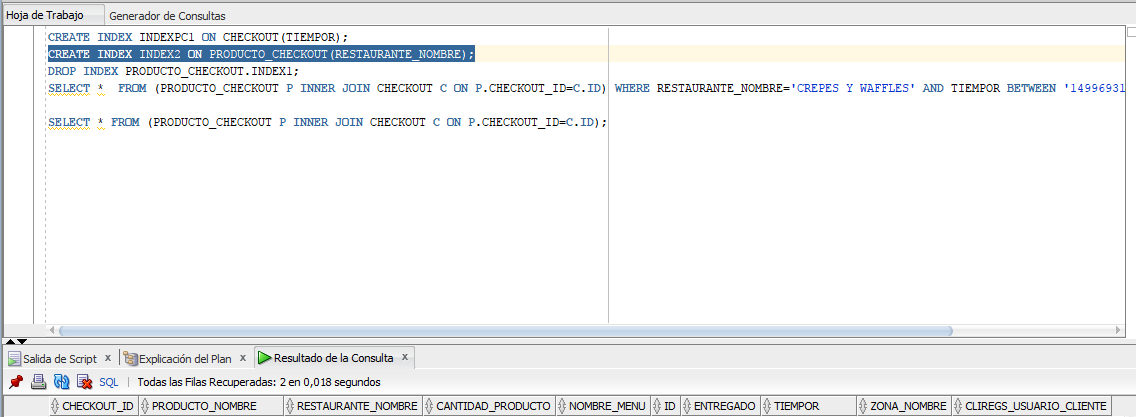
Con index 0.007 segs

Los tiempos son inconsistentes a veces se demora mas a veces menos.

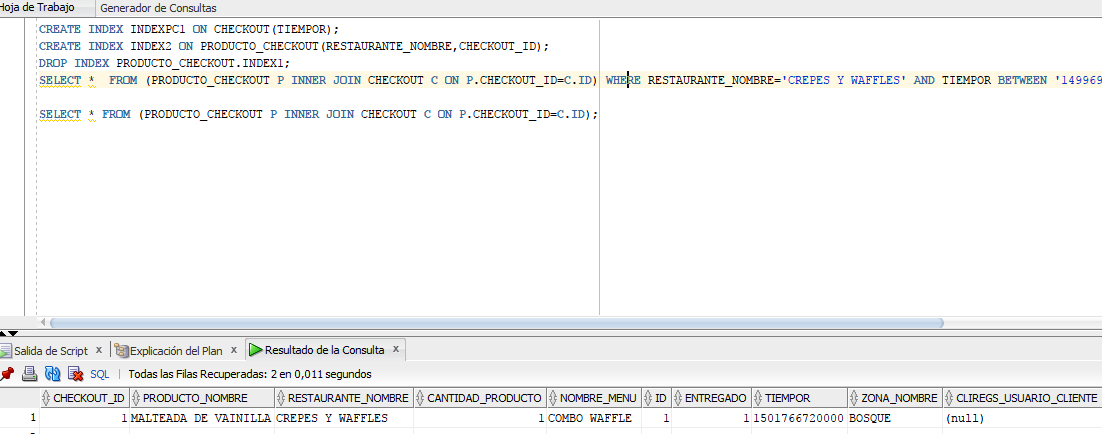
COSTOS INDEX TIEMPO



CREATE INDEX INDEX2 ON PRODUCTO\_CHECKOUT(RESTAURANTE\_NOMBRE); ES MAS SELECTIVO



El tiempo baja

 MEJORA MAS CON INDICE COMPUESTO COSTOS IGUALES.PERO APARENTEMENTE ES IGUAL DE RAPIDO SOLO CON EL ID.

Selección de índices

REQ 1 CONSULTAR CONSUMO EN ROTONDANDES:

REQ 2 CONSULTAR CONSUMO EN ROTONDANDES:

REQ 3 CONSULTAR FUNCIONAMIENTO:

REQ 4 CONSULTAR LOS BUENOS CLIENTES:

Documentación escenario de pruebas

Análisis de eficiencia