Carlos Rincón Hurtado - 201225188
David Mauricio Delgado Ruiz - 201422209
Sistemas Transaccionales - ISIS 2304

Iteración 1 - Diseño y Optimización de Consultas

ANÁLISIS DEL MODELO DEL MUNDO

Para esta iteración no tuvimos que hacer cambios al modelo del mundo, pues este se acomoda a los requerimientos de las nuevas consultas.

DISEÑO DE LA APLICACIÓN

Análisis

No tuvimos que introducir mayores cambios en el modelo relacional. Para el requerimiento de consulta de ingreso y salida de buques, agregamos un nuevo atributo opcional -es decir, puede ser null- al objeto que representa un registro de ingreso y salida de buques en la aplicación, esto con el fin de devolver los resultados de búsqueda dada las características del requerimiento.

Diseño Físico

Índices

Para el requerimiento RFC7

Índice 1
Columnas buque.id
Tipo B+
Justificación del índice
El índice fue creado automáticamente por Oracle en el momento de crear la tabla. La presencia de este índice facilita la realización del JOIN con INGRESO_SALIDA_BUQUES, pues este se hace por el id del buque.

Índice 2
Columnas INGRESO_SALIDA_BUQUES.id_buque
Tipo B+
Justificación del índice
El índice fue creado por nosotros para este requerimiento. La presencia de este índice facilita la realización del JOIN con BUQUE, pues esta tabla tiene la información del tipo y nombre de buque, necesaria para devolver un resultado al cliente.
Índice 3
Columnas INGRESO_SALIDA_BUQUES.fechaIngreso
Tipo B+

Justificación del índice
El índice fue creado creado por nosotros para este requerimiento. Dada la baja selectividad de las fechas (1/2000) La presencia de este índice facilita la realización de la verificación del AND de INGRESO_SALIDA_BUQUES, pues permite verificar las fechas muy rápido.

Para el requerimiento RFC8

Índice 1
Columnas BUQUE.ID
Tipo B+
Justificación del índice
El índice fue creado automáticamente por Oracle en el momento de crear la tabla. La presencia de este índice facilita la búsqueda de los registros de la actividad de movimiento de buques, pues se accede a estos registros de forma rapida al implementar el indice.

Índice 2
Columnas BUQUE.TIPO
Tipo HASH
Justificación del índice
Este índice aumenta la eficiencia de la consulta al acceder a los registros por TIPO de forma rapida, al crear el índice sobre este atributo el tiempo de consulta se disminuye.

Para el requerimiento RFC9
Índice 1
Columnas CARGA.ID
Tipo B+
Justificación del índice
El índice fue creado automáticamente por Oracle en el momento de crear la tabla. La presencia de este índice facilita la realización del JOIN con INGRESO_SALIDA_AREAS, pues este se hace por el id de la carga, con el fin de identificar su tipo y volumen.

Para el requerimiento RFC10

Índice 1
Columnas AREA.ID
Tipo HASH
Justificación del índice
El índice fue creado automáticamente por Oracle en el momento de crear la tabla. La presencia de este índice facilita la búsqueda de los registros de la actividad de movimiento de

carga que involucran las áreas dadas por parámetro, pues se accede a estos registros de forma rapida al implementar el indice.

Índice 2
Columnas CARGA.TIPO
Tipo HASH
Justificación del índice
Este índice aumenta la selectividad de la consulta al especificar con más parametros la búsqueda, al crear el indice sobre este atributo el tiempo de consulta se disminuye.

Índices creados automáticamente por Oracle

Antes de desarrollar la aplicación

	INDEX_NAME	INDEX_TYPE	TABLE_OWNER	TABLE_NAME	TABLE_TYPE	UNIQUENESS	COMPRESSION	PREFIX_LENGTH	TABLESPACE_NAME	INI_TRANS	MA
1	AREA_PK	NORMAL	ISIS2304B101610	AREA	TABLE	UNIQUE	DISABLED	(null)	TBSPROD	2	
2	BUQUE_PK	NORMAL	ISIS2304B101610	BUQUE	TABLE	UNIQUE	DISABLED	(null)	TBSPROD	2	
3	CARGA_PK	NORMAL	ISIS2304B101610	CARGA	TABLE	UNIQUE	DISABLED	(null)	TBSPROD	2	
4	CONTENEDOR_PK	NORMAL	ISIS2304B101610	CONTENEDOR	TABLE	UNIQUE	DISABLED	(null)	TBSPROD	2	
5	CUARTO_FRIO_PK	NORMAL	ISIS2304B101610	CUARTO_FRIO	TABLE	UNIQUE	DISABLED	(null)	TBSPROD	2	
6	DESTINOS_BUQUES_PK	NORMAL	ISIS2304B101610	DESTINOS_BUQUES	TABLE	UNIQUE	DISABLED	(null)	TBSPROD	2	
7	EQUIPO_PK	NORMAL	ISIS2304B101610	EQUIPO	TABLE	UNIQUE	DISABLED	(null)	TBSPROD	2	
8	EXPORTADOR_PK	NORMAL	ISIS2304B101610	EXPORTADOR	TABLE	UNIQUE	DISABLED	(null)	TBSPROD	2	
9	GRANEL_PK	NORMAL	ISIS2304B101610	GRANEL	TABLE	UNIQUE	DISABLED	(null)	TBSPROD	2	
10	IMPORTADOR_PK	NORMAL	ISIS2304B101610	IMPORTADOR	TABLE	UNIQUE	DISABLED	(null)	TBSPROD	2	
11	IND_BEBEDORES_ID	NORMAL	ISIS2304B101610	BEBEDORES	TABLE	NONUNIQUE	DISABLED	(null)	TBSPROD	2	
12	MULTIPROPOSITO_PK	NORMAL	ISIS2304B101610	MULTIPROPOSITO	TABLE	UNIQUE	DISABLED	(null)	TBSPROD	2	
13	PERSONA_PK	NORMAL	ISIS2304B101610	PERSONA	TABLE	UNIQUE	DISABLED	(null)	TBSPROD	2	
14	PK_BODEGA	NORMAL	ISIS2304B101610	BODEGA	TABLE	UNIQUE	DISABLED	(null)	TBSPROD	2	
15	PK_CAMION	NORMAL	ISIS2304B101610	CAMION	TABLE	UNIQUE	DISABLED	(null)	TBSPROD	2	
16	PK_CARGA_USA_EQUIPO	NORMAL	ISIS2304B101610	CARGA_USA_EQUIPO	TABLE	UNIQUE	DISABLED	(null)	TBSPROD	2	
17	PK_COBERTIZO	NORMAL	ISIS2304B101610	COBERTIZO	TABLE	UNIQUE	DISABLED	(null)	TBSPROD	2	
18	PK_CRB	NORMAL	ISIS2304B101610	CARGA_RESERVADA_BUQUE	TABLE	UNIQUE	DISABLED	(null)	TBSPROD	2	
19	PK_FACTURA	NORMAL	ISIS2304B101610	FACTURA	TABLE	UNIQUE	DISABLED	(null)	TBSPROD	2	
20	PK_ISA	NORMAL	ISIS2304B101610	INGRESO_SALIDA_AREA	TABLE	UNIQUE	DISABLED	(null)	TBSPROD	2	

Se puede ver que Oracle creó índices para todas las llaves primarias de la base de datos. Por defecto, estos índices son B+.
Después de desarrollar la aplicación

Análisis

Análisis de los requerimientos funcionales

Para el requerimiento RFC7

Sentencia SQL que responde el requerimiento
SELECT ISB.ID_BUQUE AS ID_BUQUE, B.NOMBRE AS NOMBRE, ISB.FECHA_INGRESO AS FECHA_INGRESO, ISB.FECHA_SALIDA AS FECHA_SALIDA, B.NOMBRE AS NOMBRE_BUQUE, , B.TIPO_BUQUE AS TIPO_BUQUE, FROM INGRESO_SALIDA_BUQUES ISB INNER JOIN BUQUE B
WHERE ISB.FECHA_INGRESO >= TO_DATE('fechaInicioSql', 'YYYY-MM-DD')
AND

ISB.FECHA_SALIDA <= TO_DATE('fechaFinSql ', 'YYYY-MM-DD')
AND B.NOMBRE = nombreBuque
AND B.TIPO_BUQUE = tipoDeBuque

Distribución de los Datos - Análisis del tamaño de respuesta

En promedio, un barco puede ingresar cada dos o 3 días. Además de eso, se sabe dada la manera como se poblaron las tablas que em promedio ingresan 500 barcos cada dos días. Esto quiere decir que para unos parámetros de búsqueda, dado por ejemplo 2 días y el TIPO RORO, tendremos apróximente ⅓ de 500, 165 registros más o menos, y obtenemos 162, una buena aproximación

Valores del análisis relevantes en la ejecución del plan

Los valores relevantes son las fechas de ingreso y salida del puerto, pues estas determinan el tamaño de la respuesta.

Planes de consulta obtenidos de Oracle

Tiempos obtenidos para la consulta

0.083 segundos

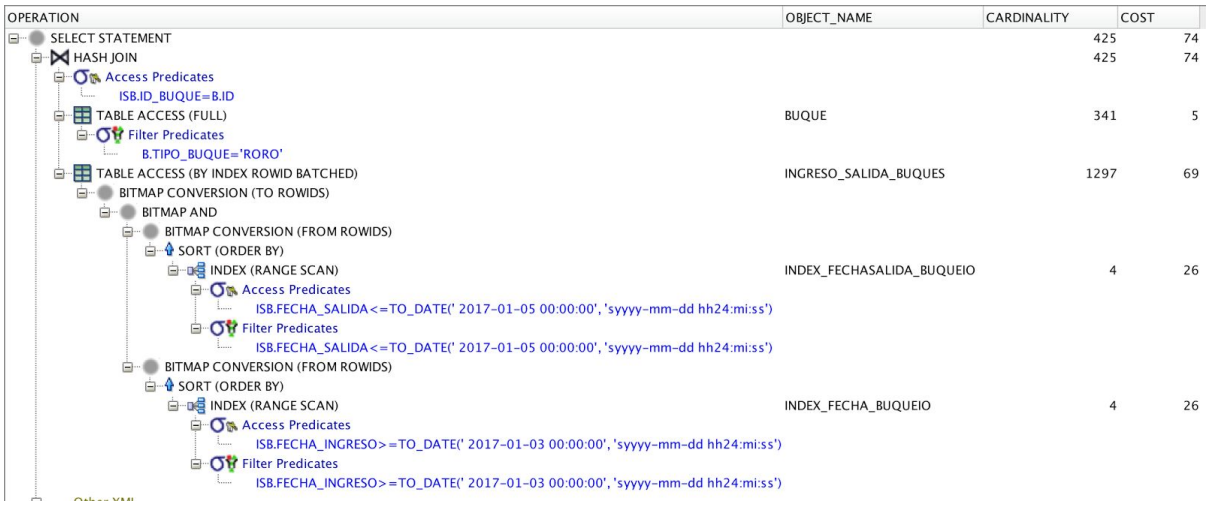
Escenario de análisis de desempeño

Después de agregar los dos índices mencionado arriba el costo se redujo notablemente, de 1133 a sólo 74 unidades. Esto es una reducción a un costo al menos 100 veces menor, lo que hace que esta consulta que requiere el cliente sea muy eficiente.

Plan de consulta propuesto

Yo agregaría otro índice a la base de datos sobre la fechaSALIDA, que permitiera hacer que este acceso fuera mucho más barato, pues es lo que más le da costo en unidades a la consulta. Dada la manera como fueron llenados los datos, se sabe que en promedio suceden 500 ingresos/egresos cada día, lo que nos da un cociente de 500/1000000 = 5/10000 = 1/2000, lo cual tiene una selectividad alta.

Ahora bien, al agregarlo, se obtiene lo siguiente:



OPERATION	OBJECT_NAME	CARDINALITY	COST
SELECT STATEMENT		425	74
HASH JOIN		425	74
Access Predicates			
ISB.ID_BUQUE=B.ID			
TABLE ACCESS (FULL)	BUQUE	341	5
Filter Predicates			
B.TIPO_BUQUE='RORO'			
TABLE ACCESS (BY INDEX ROWID BATCHED)	INGRESO_SALIDA_BUQUES	1297	69
BITMAP CONVERSION (TO ROWIDS)			
BITMAP AND			
BITMAP CONVERSION (FROM ROWIDS)			
SORT (ORDER BY)			
INDEX (RANGE SCAN)	INDEX_FECHASALIDA_BUQUEIO	4	26
Access Predicates			
ISB.FECHA_SALIDA<=TO_DATE(' 2017-01-05 00:00:00', 'syyyy-mm-dd hh24:mi:ss')			
Filter Predicates			
ISB.FECHA_SALIDA<=TO_DATE(' 2017-01-05 00:00:00', 'syyyy-mm-dd hh24:mi:ss')			
BITMAP CONVERSION (FROM ROWIDS)			
SORT (ORDER BY)			
INDEX (RANGE SCAN)	INDEX_FECHA_BUQUEIO	4	26
Access Predicates			
ISB.FECHA_INGRESO>=TO_DATE(' 2017-01-03 00:00:00', 'syyyy-mm-dd hh24:mi:ss')			
Filter Predicates			
ISB.FECHA_INGRESO>=TO_DATE(' 2017-01-03 00:00:00', 'syyyy-mm-dd hh24:mi:ss')			

Comparación entre planes

Como se puede ver en las dos imágenes de arriba, la introducción del nuevo índice que se pensaba reduciría el costo, aunque sí cambió el plan de ejecución, ahora usando Bitmaps y haciendo un par de ordenamientos, mantuvo el costo igual al plan anterior. Debido a esto, decidimos eliminar ese índice.

Resultado de la consulta

	ID_BUQUE	NOMBRE	FECHA_INGRESO	FECHA_SALIDA	TIPO_BUQUE
138	1666	BUQ1666	03/01/17	05/01/17	RORO
139	1688	BUQ1688	03/01/17	05/01/17	RORO
140	1913	BUQ1913	03/01/17	05/01/17	RORO
141	1279	BUQ1279	03/01/17	05/01/17	RORO
142	1500	BUQ1500	03/01/17	05/01/17	RORO
143	1188	BUQ1188	03/01/17	05/01/17	RORO
144	1511	BUQ1511	03/01/17	05/01/17	RORO
145	1404	BUQ1404	03/01/17	05/01/17	RORO
146	1867	BUQ1867	03/01/17	05/01/17	RORO

Para el requerimiento RFC8

Sentencia SQL que responde el requerimiento

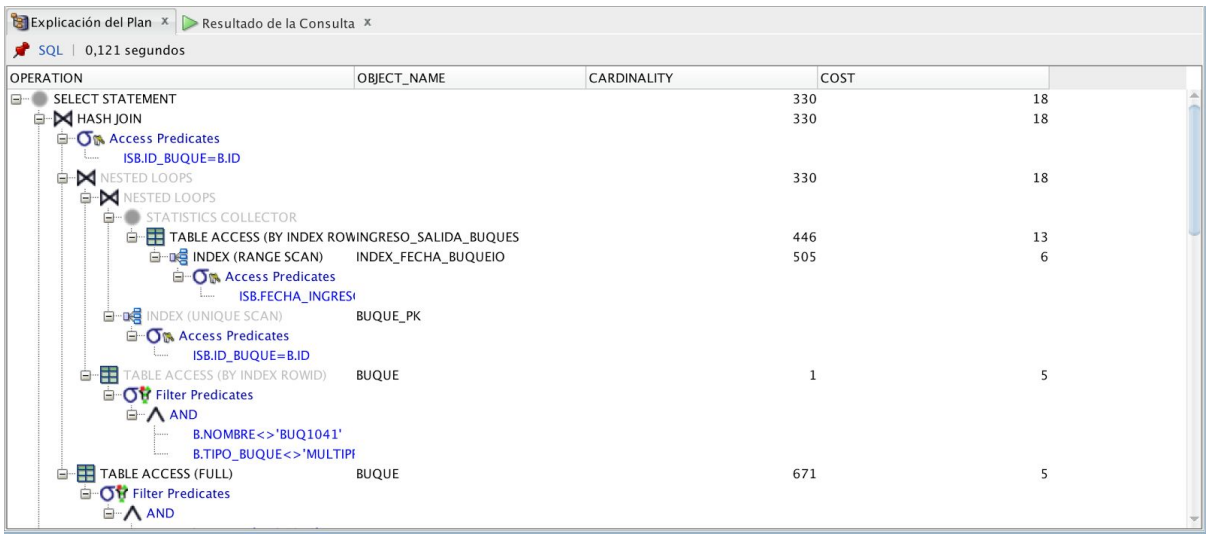
```
SELECT ISB.ID_BUQUE AS ID_BUQUE, B.NOMBRE AS NOMBRE, ISB.FECHA_INGRESO AS
FECHA_INGRESO, ISB.FECHA_SALIDA AS FECHA_SALIDA, B.NOMBRE AS
NOMBRE_BUQUE , B.TIPO_BUQUE AS TIPO_BUQUE
FROM INGRESO_SALIDA_BUQUES ISB INNER JOIN BUQUE B ON ISB.ID_BUQUE=B.ID
WHERE ISB.FECHA_INGRESO >=  TO_DATE('2017-01-03', 'YYYY-MM-DD')
AND ISB.FECHA_SALIDA <=  TO_DATE('2017-03-28', 'YYYY-MM-DD')
AND B.NOMBRE <> 'BUQ1041'
```

AND B.TIPO_BUQUE <> 'MULTIPROPOSITO';

Distribución de los Datos - Análisis del tamaño de respuesta
El tamaño de la respuesta depende directamente del rango de fechas definido en la consulta, a nivel de prueba se trabaja con un rango de 3 meses el cual arroja en promedio 14.000 resultados.

Valores del análisis relevantes en la ejecución del plan
Los valores relevantes incluyen la cantidad de movimientos de buques generados en el puerto.

Planes de consulta obtenidos de Oracle

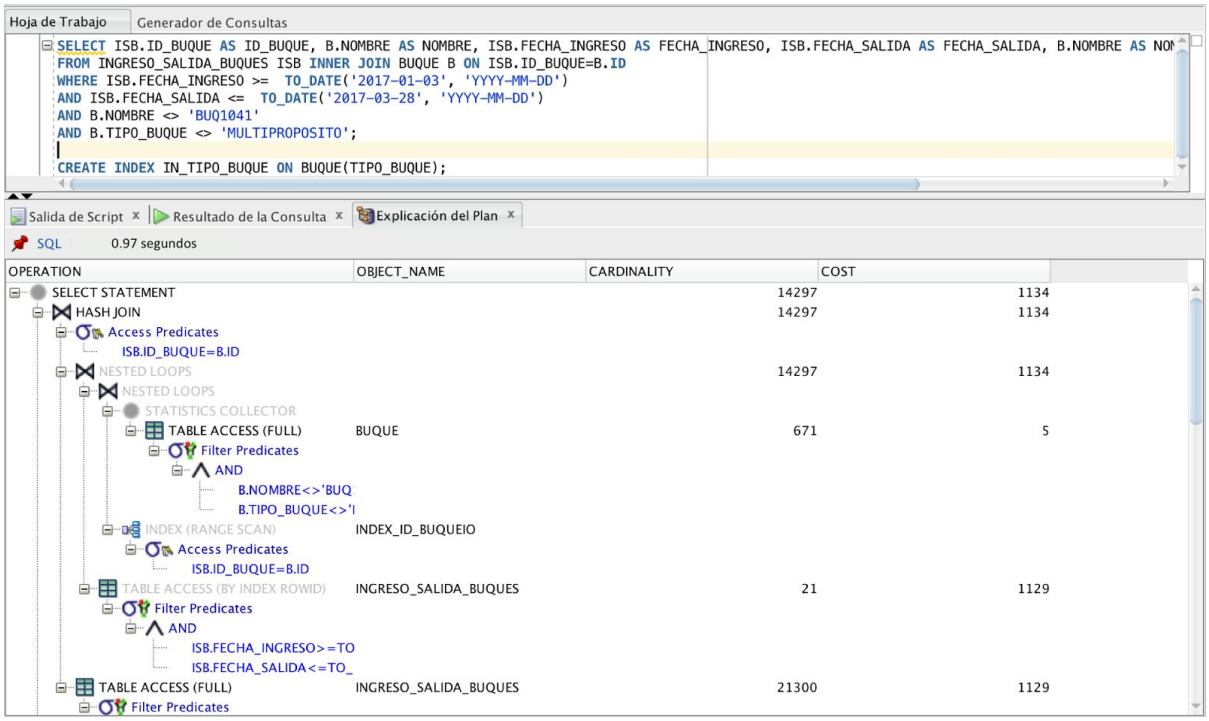


Tiempos obtenidos para la consulta
0.121 segundos

Escenario de análisis de desempeño
Al implementar HASH JOIN se hace el acceso a los datos mediante un índice, siendo esto una consulta eficiente en tiempo de búsqueda.

Plan de consulta propuesto

Se espera que que al crear el índice sobre TIPO_BUQUE se mejor el tiempo de búsqueda



Comparación entre planes

Podemos observar que después de la creación del índice el tiempo de búsqueda pasó de 0.121 a 0.97 segundos.

Resultado de la consulta

	ID_BUQUE	NOMBRE	FECHA_INGRESO	FECHA_SALIDA	NOMBRE_BUQUE	TIPO_BUQUE
1	1027	BUQ1027	2017-01-31	2017-02-02	BUQ1027	PORTACONTENEDOR
2	1493	BUQ1493	2017-01-31	2017-02-02	BUQ1493	RORO
3	1451	BUQ1451	2017-01-31	2017-02-02	BUQ1451	RORO
4	1960	BUQ1960	2017-01-31	2017-02-02	BUQ1960	RORO
5	1713	BUQ1713	2017-01-31	2017-02-02	BUQ1713	PORTACONTENEDOR
6	1076	BUQ1076	2017-01-31	2017-02-02	BUQ1076	RORO
7	1690	BUQ1690	2017-01-31	2017-02-02	BUQ1690	PORTACONTENEDOR
8	1230	BUQ1230	2017-01-31	2017-02-02	BUQ1230	PORTACONTENEDOR
9	1995	BUQ1995	2017-01-31	2017-02-02	BUQ1995	RORO
10	1052	BUQ1052	2017-01-31	2017-02-02	BUQ1052	PORTACONTENEDOR
11	1842	BUQ1842	2017-01-31	2017-02-02	BUQ1842	RORO
12	1994	BUQ1994	2017-01-31	2017-02-02	BUQ1994	PORTACONTENEDOR
13	1564	BUQ1564	2017-01-31	2017-02-02	BUQ1564	RORO
14	1444	BUQ1444	2017-01-31	2017-02-02	BUQ1444	PORTACONTENEDOR
15	1753	BUQ1753	2017-01-31	2017-02-02	BUQ1753	PORTACONTENEDOR
16	1395	BUQ1395	2017-01-31	2017-02-02	BUQ1395	PORTACONTENEDOR
17	1344	BUQ1344	2017-01-31	2017-02-02	BUQ1344	RORO
18	1984	BUQ1984	2017-01-31	2017-02-02	BUQ1984	RORO
19	1096	BUQ1096	2017-01-31	2017-02-02	BUQ1096	PORTACONTENEDOR

Para el requerimiento RFC9

Sentencia SQL que responde el requerimiento

```
SELECT CAR.ID, CAR.TIPO, CAR.FECHA_ORDEN, CAR.ORIGEN, CAR.DESTINO,
ISA.ID_IMPORTADOR, CAR.ID_EXPORTADOR
FROM CARGA CAR
INNER JOIN
INGRESO_SALIDA_AREA ISA
ON CAR.ID = ISA.ID_CARGA
WHERE CAR.TIPO = 'GRANEL'
AND CAR.VOLUMEN > 300;
```

Distribución de los Datos - Análisis del tamaño de respuesta

Dado que en promedio 1 de cada 3 cargas es granel, los resultados son bastante grandes. Además, hay caso 2 millones de registros, de esos, al parecer más de 2000 cumplen con las condiciones mencionadas

Valores del análisis relevantes en la ejecución del plan

Los valores relevantes incluyen la cantidad de cargas del tipo suministrado por parámetro que ingresan al puerto, y el volumen de las cargas del puerto.

Planes de consulta obtenidos de Oracle

Tiempos obtenidos para la consulta

0.133

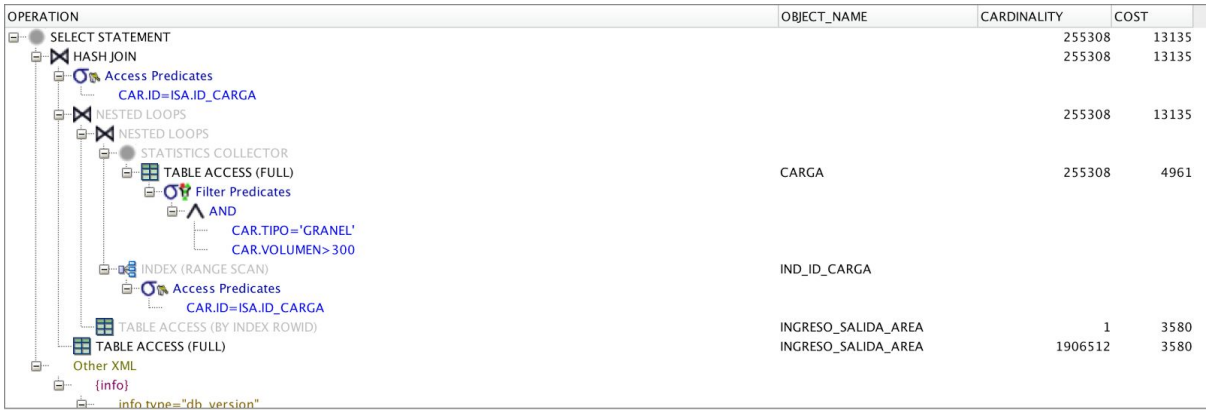
Escenario de análisis de desempeño

El desempeño de este plan no es muy eficiente, pues debe hacer un acceso completo a toda la tabla.

Plan de consulta propuesto y Comparación entre planes

Ahora bien, si analizamos las columnas y campos importantes, nos damos cuenta que ninguno de estos tiene una selectividad apropiada para el uso de un índice, lo que hace que la consulta no pueda mejorarse en estos aspectos. Ahora bien, si entrara otro parámetro, como la fecha en que se da la orden, esa si podría hacer uso de un índice y si mejoraría el costo de la transacción.

Podría introducirse un índice sobre ID_CARGA en INGRESO_SALIDA_AREA, pero ese no mejora en nada el rendimiento, como se puede ver a continuación:



En base en esto, decidimos eliminar el índice.
Resultado de la consulta

	ID	TIPO	FECHA_ORDEN	ORIGEN	DESTINO	ID_IMPORTADOR	ID_EXPORTADOR
1	1688	GRANEL	03/01/17	ORIGEN4020	PUERTOANDES	1149	1324
2	1707	GRANEL	03/01/17	ORIGEN3019	PUERTOANDES	1013	1305
3	1710	GRANEL	03/01/17	ORIGEN3366	PUERTOANDES	1232	1332
4	1711	GRANEL	03/01/17	ORIGEN1614	PUERTOANDES	1040	1470
5	1716	GRANEL	03/01/17	ORIGEN2295	PUERTOANDES	1140	1331
6	1727	GRANEL	03/01/17	ORIGEN3177	PUERTOANDES	1024	1495
7	1735	GRANEL	03/01/17	ORIGEN3658	PUERTOANDES	1243	1422
8	1742	GRANEL	03/01/17	ORIGEN5646	PUERTOANDES	1248	1351
9	1743	GRANEL	03/01/17	ORIGEN4939	PUERTOANDES	1070	1483
10	1752	GRANEL	03/01/17	ORIGEN5544	PUERTOANDES	1003	1263
11	1763	GRANEL	03/01/17	ORIGEN5423	PUERTOANDES	1246	1277
12	1765	GRANEL	03/01/17	ORIGEN1311	PUERTOANDES	1007	1259

Para el requerimiento RFC10

Sentencia SQL que responde el requerimiento

```
SELECT ISA.ID_CARGA, ISA.ID_AREA, ISA.VIENE_DE, ISA.FECHA_INGRESO,
ISA.FECHA_SALIDA, C.TIPO
FROM INGRESO_SALIDA_AREA ISA JOIN CARGA C ON ISA.ID_CARGA=C.ID
WHERE ISA.ID_AREA=1272;
```

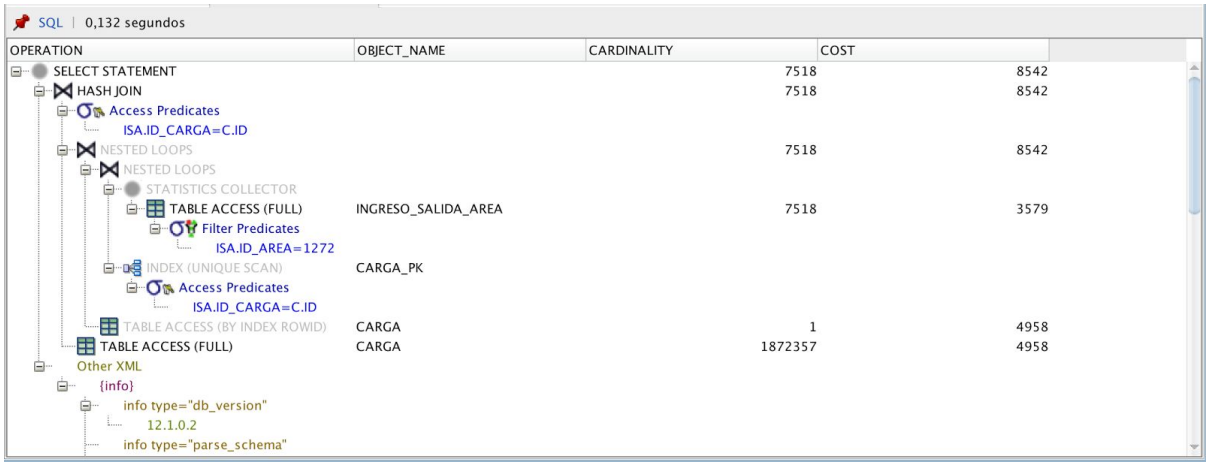
Distribución de los Datos - Análisis del tamaño de respuesta

Por ID de Área se obtienen en promedio 8.000 registros los cuales representan movimientos de carga que involucran esta área, esto corresponde a 1/250 de los datos.

Valores del análisis relevantes en la ejecución del plan

Los valores relevantes incluyen la cantidad de movimiento de cargas generadas en el puerto.

Planes de consulta obtenidos de Oracle


SQL | 0,132 segundos

OPERATION	OBJECT_NAME	CARDINALITY	COST
SELECT STATEMENT		7518	8542
HASH JOIN		7518	8542
Access Predicates			
ISA.ID_CARGA=C.ID			
NESTED LOOPS		7518	8542
NESTED LOOPS			
STATISTICS COLLECTOR			
TABLE ACCESS (FULL)	INGRESO_SALIDA_AREA	7518	3579
Filter Predicates			
ISA.ID_AREA=1272			
INDEX (UNIQUE SCAN)	CARGA_PK		
Access Predicates			
ISA.ID_CARGA=C.ID			
TABLE ACCESS (BY INDEX ROWID)	CARGA	1	4958
TABLE ACCESS (FULL)	CARGA	1872357	4958
Other XML			
{info}			
info type="db_version"			
12.1.0.2			
info type="parse_schema"			

Tiempos obtenidos para la consulta

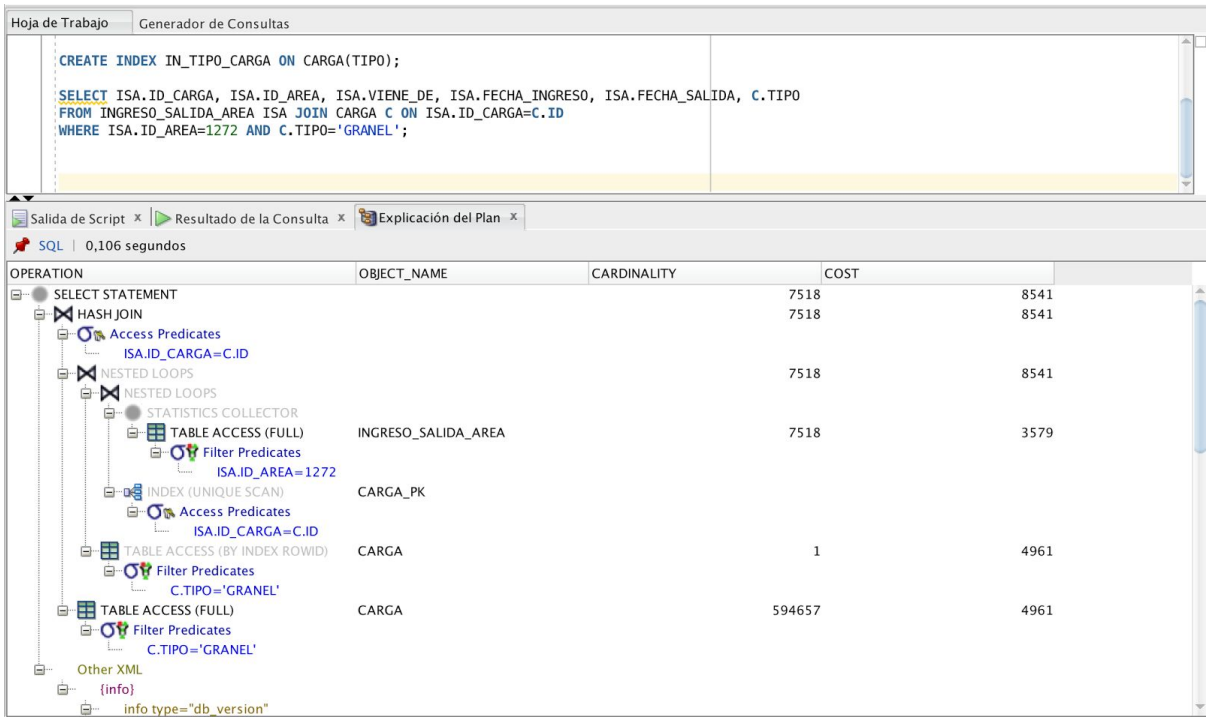
0.132 segundos

Escenario de análisis de desempeño

Esta búsqueda resulta eficiente pues al utilizar HASH JOIN como acceso a los datos el tiempo de búsqueda se reduce notablemente pues no es necesario recorrer toda la tabla.

Plan de consulta propuesto

Aparte del índice por defecto generado por Oracle al generar la tabla, no existen más atributos como parámetro de búsqueda que se pueda utilizar como índice, pues el único criterio de búsqueda es por ID. Ahora si se buscara reducir el tamaño de la respuesta del requerimiento esto se podría hacer filtrando la búsqueda, además, por tipo de carga, lo cual se espera reduzca el tiempo de búsqueda.



Comparación entre planes

Podemos observar que después de la creación del índice el tiempo de búsqueda pasó de 0.132 a 0.106 segundos.

Resultado de la consulta

Salida de Script | Resultado de la Consulta | Explicación del Plan

SQL | Todas las Filas Recuperadas: 2703 en 2,406 segundos

	ID_CARGA	ID_AREA	VIENE_DE	FECHA_INGRESO	FECHA_SALIDA	TIPO
1	1920	1272	BARCO	2017-01-03	2017-01-05	GRANEL
2	1948	1272	BARCO	2017-01-03	2017-01-05	GRANEL
3	3675	1272	BARCO	2017-01-03	2017-01-05	GRANEL
4	3688	1272	BARCO	2017-01-03	2017-01-05	GRANEL
5	3848	1272	BARCO	2017-01-03	2017-01-05	GRANEL
6	4153	1272	BARCO	2017-01-03	2017-01-05	GRANEL
7	4706	1272	BARCO	2017-01-03	2017-01-05	GRANEL
8	5036	1272	BARCO	2017-01-03	2017-01-05	GRANEL
9	5681	1272	BARCO	2017-01-03	2017-01-05	GRANEL
10	6031	1272	BARCO	2017-01-05	2017-01-07	GRANEL
11	7684	1272	BARCO	2017-01-05	2017-01-07	GRANEL
12	8091	1272	BARCO	2017-01-05	2017-01-07	GRANEL
13	10014	1272	BARCO	2017-01-05	2017-01-07	GRANEL
14	10813	1272	BARCO	2017-01-05	2017-01-07	GRANEL
15	11027	1272	BARCO	2017-01-07	2017-01-09	GRANEL
16	11425	1272	BARCO	2017-01-07	2017-01-09	GRANEL
17	13186	1272	BARCO	2017-01-07	2017-01-09	GRANEL
18	13929	1272	BARCO	2017-01-07	2017-01-09	GRANEL
19	22483	1272	BARCO	2017-01-11	2017-01-13	GRANEL

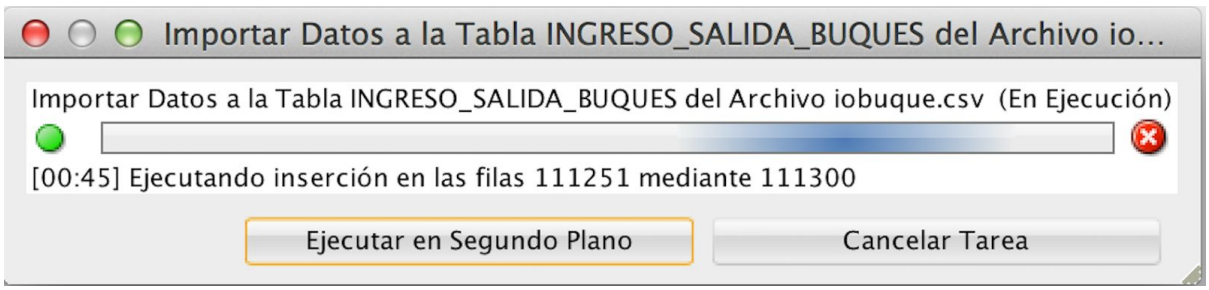
CONSTRUCCIÓN DE LA APLICACIÓN Y ANÁLISIS DE RESULTADOS

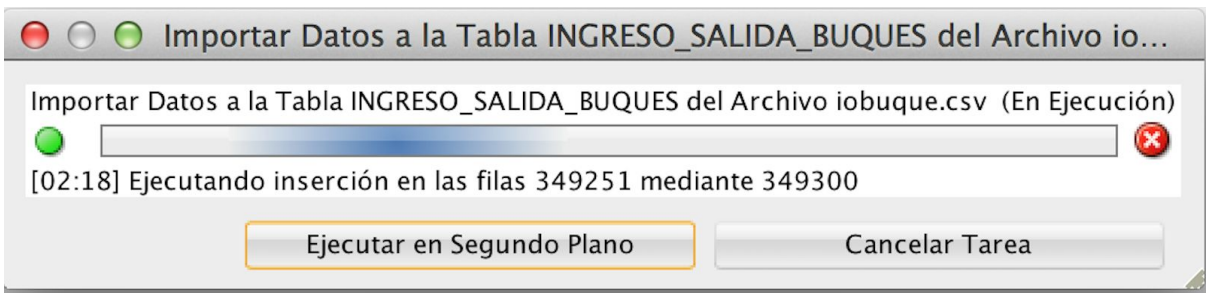
Documentación del proceso de carga de datos

Estudiamos varias opciones al momento de decidir cómo íbamos a poblar la base de datos con muchos registros, - 1 millón o más- . Estas fueron las opciones que encontramos:

1. Usar un archivo CSV generado por un generador aleatorio de datos en línea y SQL Loader para para cargarlo. Esta opción aunque viable, requería la instalación de herramientas y conectarse a la base de datos usando SSH. El proceso de configuración demoraría bastante, pues las herramientas tienen tutoriales para Windows y no para Mac, sistema operativo que mi compañero y yo tenemos.
2. Usar un script de SQL. Esta solución es bastante rápida, pues no necesita de la conexión con nuestro computador para recibir los archivos. Sin embargo, dada la complejidad de las relaciones entre nuestras tablas para hacer este script necesitamos de conocimientos un poco extensos de PL/SQL, y por tanto, fue descartado.
3. Usando nuestro lenguaje de programación favorito, escribir un archivo CSV y luego importar usando la herramienta de importación por defecto de SQL Developer. Esta fue la opción que escogimos, pues el nivel de flexibilidad de un lenguaje de programación normal nos permitió escribir el archivo más fácilmente.

Aquí se pueden ver imágenes del proceso de carga de una de las tablas:





Desarrollo de la capa de recursos REST

Para la capa de recursos sólo agregamos 4 nuevos métodos que toman el objeto creado por el API de JAR XS y lo envía al master para realizar la consulta.

Cambios y desarrollo de las transacciones

Ahora seguimos una mejor arquitectura, pues hacemos uso de los útiles QueryParam que una librería que implementa una arquitectura REST tiene. Esto permite mayor claridad en el código.

Cambios en los DAO

No hay mayores cambios en los DAO's, sólo los métodos que realizan los nuevos requerimientos

Análisis del proceso de optimización y el modelo de ejecución de consultas

En conclusión, Oracle sí que hace un buen trabajo optimizando las consultas, debido a que la posible introducción de índices obvios de los planes de consulta que propusimos fue irrelevante y el desempeño y rapidez de las consultas no cambió.

Ahora bien, si notamos que la introducción de ciertos índices, dado nuestro conocimiento de la distribución de los datos sí redujo de manera significante.