1 (1 %) Análisis Ajuste el modelo del mundo (modelo conceptual: diagrama de clases UML) propuesto en la iteración anterior, si lo requiere. Indique cuáles clases del modelo del mundo fueron actualizadas o creadas en esta iteración.

Disponible en la carpeta docs del proyecto java*

# 2 Diseño de la aplicación

- ✓ (1%) A partir del diseño existente, analice el impacto que representa la introducción de los nuevos requerimientos y restricciones a nivel del modelo conceptual. Realice los cambios necesarios en su modelo relacional para respetar las reglas de negocio y asegurar la calidad del mismo. Tenga en cuenta los comentarios recibidos en la sustentación de los talleres anteriores. Documente el diseño y las decisiones tomadas para crear los elementos de la base de datos que da el respaldo de persistencia a la aplicación, a partir del modelo conceptual. ⟩ Sea claro en mencionar explícitamente los cambios relevantes entre su diseño entregado en iteraciones anteriores y este.

    - Debido a la manera como estamos manejando los datos en la base de datos no fu necesario ningún cambio en la base de datos en cuanto a creación de tablas, debimos crear índices para poder asegurar que nuestras consultas cumplan el límite de respuesta de 0.8 segundos

(63 %) Diseño físico. Analice la aplicación completa resultante de la iteración anterior y de los nuevos requerimientos para realizar el diseño físico correspondiente. En particular, diseñe los índices necesarios para el adecuado rendimiento global de la aplicación. ϖ

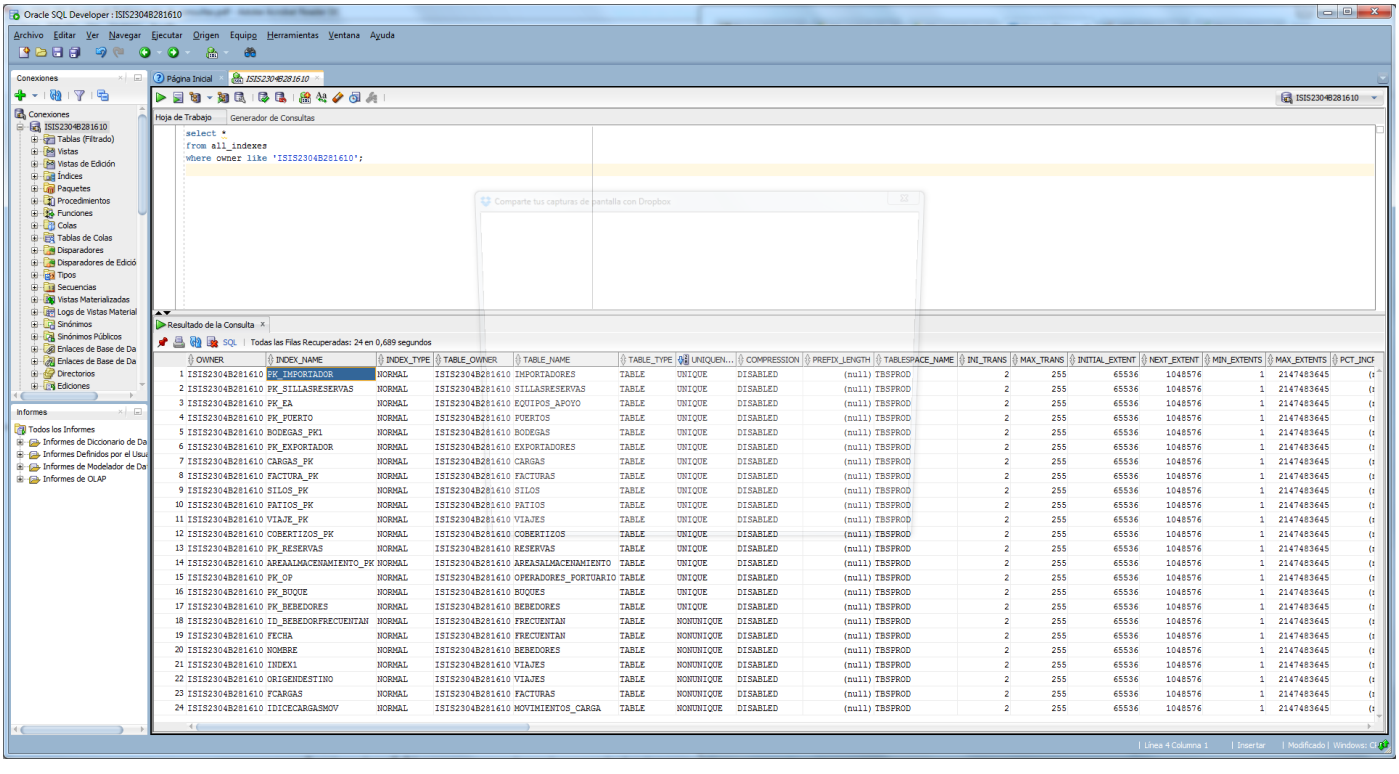
- (19%) Documente su diseño físico.
    - Justifique la selección de índices desde el punto de vista de cada uno de los requerimientos funcionales. Indique claramente cuál es el tipo de índice utilizado (B+, Hash, …, primario, secundario) y tenga en cuenta el costo de almacenamiento y mantenimiento asociado a los índices.

- En el momento de pensar en que tipos de índices y en que columnas lo haríamos decidimos investigar un poco más para así poder tomar la decisión correcta, Para que los índices tengan un impacto relevante en el tiempo y la complejidad de las consultas estos se deben crear en las columnas que sean utilizadas en las consultas y sabiendo que la generación de los mismos tendrá un impacto positivo sobre la base de datos y las consultas generadas en la misma, por ejemplo si en una búsqueda nos piden los viajes que salen de Puerto Andes si su buque asociado es X, no se pondrá un índice en fecha de salida, ya que, esto no generaría ningún cambio positivo en el tiempo ni en la complejidad de la consulta, es más podría aumentar el costo de la consulta.

  Basándonos en uno de los documentos de Oracle en donde dan consejos para el manejo adecuado, decidimos que no era conveniente usar índices de tipo Mapa de bits, *"En realidad, siempre se recomienda usar índices de mapas de bits para sistemas en los que los datos no son actualizados por muchos sistemas simultáneos frecuentemente*" (Sharma, 2014). Debido a que las tablas son actualizadas, modificadas y consultadas por muchos clientes a la vez (solución Transaccional), decidimos que no debíamos usar bitmaps para estos índices. Por otro lado no vimos la necesidad de implementar índices de Dominio ya que estos se usan en casos especiales o para índices personalizados.

  "Oracle permite que las tablas sean indexadas por estructuras de índices que no sean propias de Oracle. Esta característica de extensibilidad del servidor Oracle permite a los fabricantes de software desarrollar los llamados cartuchos- Con funcionalidad para domínios de aplicación específicos, tales como texto, datos espaciales, e imágenes, con la funcionalidad de indexado mas allá de la proporcionada por los tipos de índice Oracle estándar. Para implementar la lógica para crear, mantener y buscar en el índice, el diseñador de índices debe asegurar que se adhiere a un protocolo específico en Su interacción con el servidor Oracle. Un índice de dominio se debe registrar en el diccionario de datos junto con los operadores que soporta". (ABD-UCV-Computacion, 2013). Sabiendo esto, decidimos que los índices de nuestra tabla de datos serán de tipo Único (valores que no se repiten) y No único (valores que pueden estar repetidos) y a su vez pueden ser sencillos o compuestos

- Según su modelo de datos, para los índices creados de forma automática por Oracle Γ Incluya una foto de pantalla con la información generada por Oracle asociada a los índices existentes. Γ Analice los índices encontrados. Específicamente, analice por qué fueron creados por Oracle y si ayudan al rendimiento de los requerimientos funcionales.




Listado de índices de la base de datos

Oracle por defecto crea índices Únicos basados en ArbolesB para las columnas que sean llave primaria, llamándolas igual que la restricción de la llave primaria. Por ejemplo en el caso de nuestra base de datos se llaman PK_nombreTabla.

Los índices creados por Oracle son de gran utilidad cuando se quieren hacer búsquedas por el identificador único de la tupla que usualmente se define como llave primaria, o cuando se quieren hacer búsquedas que comparen la llave primaria de una tabla con la llave foránea de otra. Debido a que este índice almacena los datos en un árbol b hace que la búsqueda sea más fácil y eficiente, ya que si estuvieran en una lista, se debería recorrer toda la lista hasta encontrar la tupla que haga “match”.

**(44%)** Documente plenamente el análisis realizado, incluyendo los siguientes aspectos **para cada requerimiento funcional solicitado**

- ✓ Documentación del escenario de pruebas

1. Sentencias SQL que responden el requerimiento y que fueron analizadas.
2. Distribución de los datos con respecto a los parámetros de entrada utilizados en el requerimiento funcional. En particular se quiere un análisis de distribución que permita ver cómo puede cambiar el tamaño de la respuesta según el valor de los parámetros utilizados y la configuración de los datos de prueba.
3. Valores de los parámetros utilizados en el análisis y que constituyen diferenciadores en los planes de ejecución obtenidos.
4. Planes de consulta obtenidos en Oracle para la ejecución del requerimiento. Para ello, documente con una foto de pantalla los planes de consulta obtenidos en SQLDevelopper.
5. Tiempos obtenidos con la ejecución de cada uno de los planes. Estos tiempos son medidos desde el núcleo de la aplicación, es decir, no incluyen la parte de interacción con el usuario, ingreso de datos ni despliegue de resultados.

- ✓ Análisis de eficiencia

6. Establezca escenarios de datos que le permitan validar diferentes selectividades.
7. Para cada requerimiento funcional, seleccione un escenario de análisis y diseñe el plan de ejecución de consulta propuesto por el grupo, de acuerdo con su conocimiento del modelo y de la aplicación.

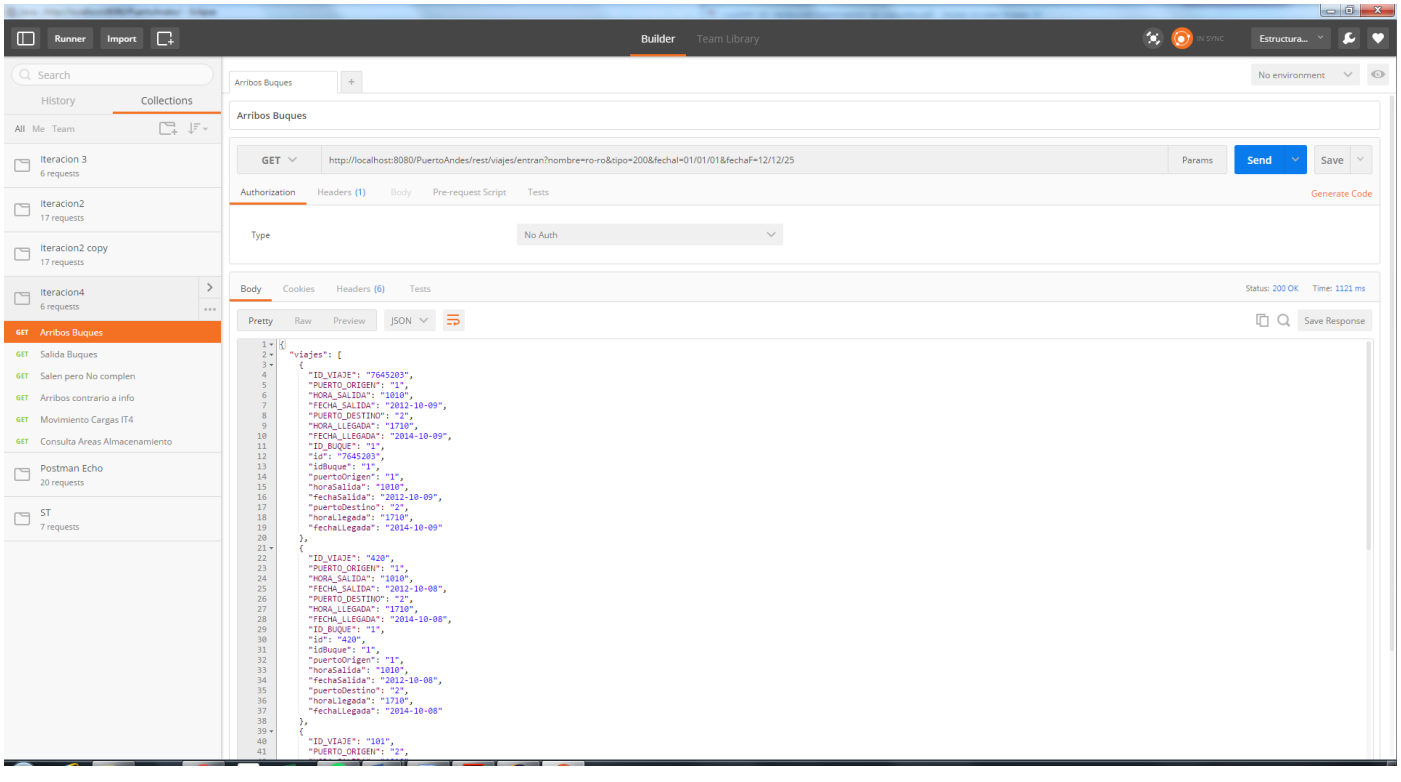
RFC7. CONSULTAR LOS ARRIBOS Y SALIDAS DE BUQUES EN PUERTOANDES – RFC1-v2

1. Sentencia SQL que fue utilizada:

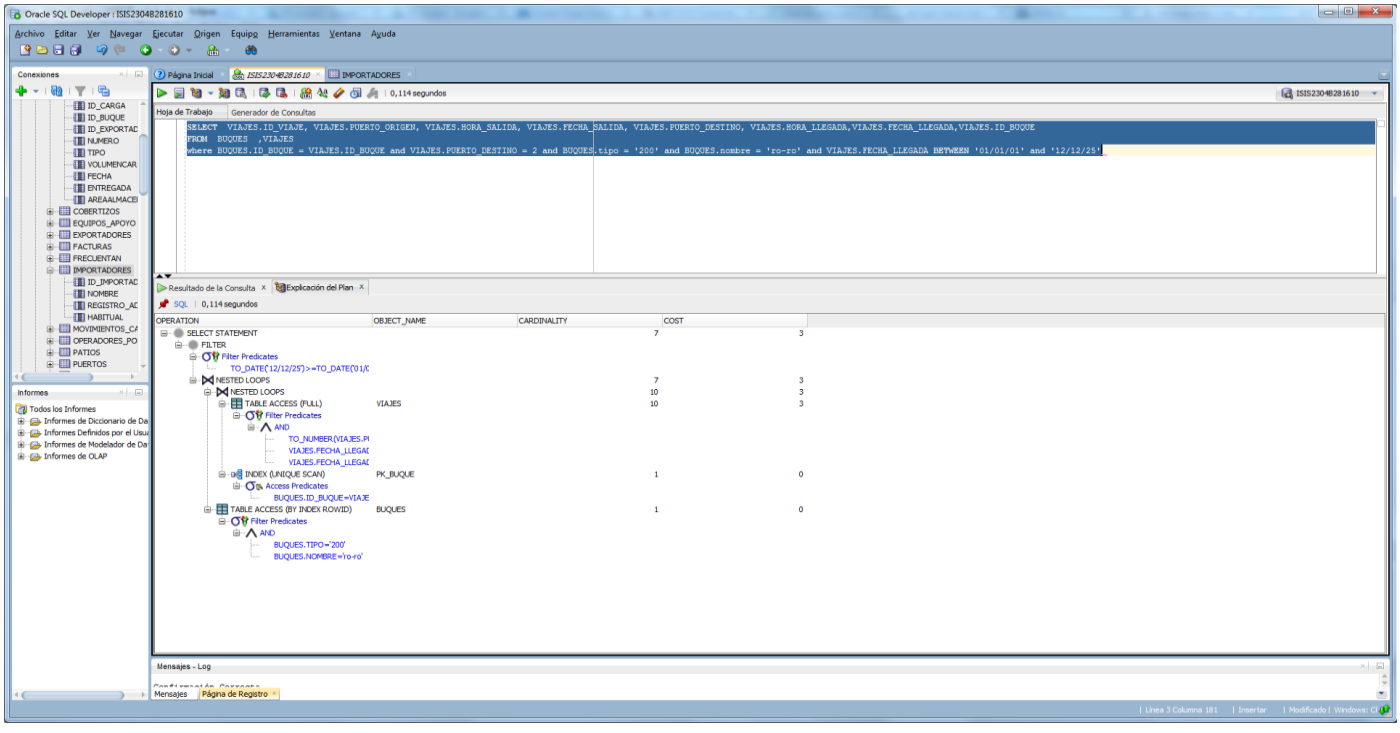
"SELECT VIAJES.ID_VIAJE, VIAJES.PUERTO_ORIGEN, VIAJES.HORA_SALIDA, VIAJES.FECHA_SALIDA, VIAJES.PUERTO_DESTINO, VIAJES.HORA_LLEGADA,VIAJES.FECHA_LLEGADA,VIAJES.ID_BUQUE FROM BUQUES ,VIAJES where BUQUES.ID_BUQUE = VIAJES.ID_BUQUE and VIAJES.PUERTO_DESTINO = 2 and BUQUES.tipo = '"+tipoBuque+"' and BUQUES.nombre = '"+nombreBuque+"' and VIAJES.FECHA_LLEGADA BETWEEN '"+fechaInicial+"' and '"+ fechaFinal+"'";

2. El resultado de la respuesta depende de muchos factores, primero que todo el tipo de buque por el cual se hace la búsqueda ya que hay buques que tienen asociados más viajes que otros, además que puede haber más de un buque con el mismo tipo o con el mismo nombre, y finalmente dependiendo del rango de fechas, si el rango es grande habrán más tuplas que cumplan con la consulta y entre más pequeño sea el rango menos tuplas cumplirán con los criterios de búsqueda.

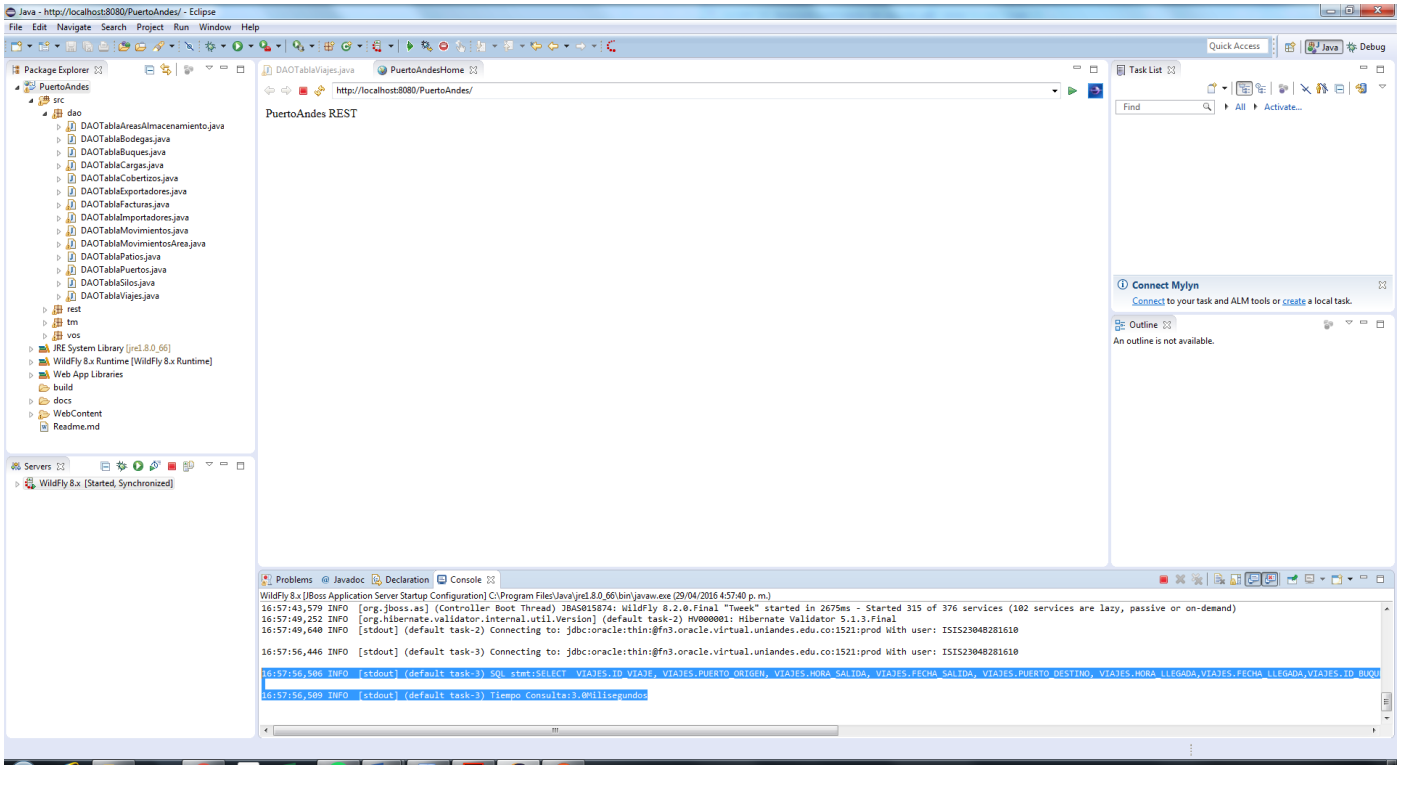
3. Para la prueba se utilizaron los valores:



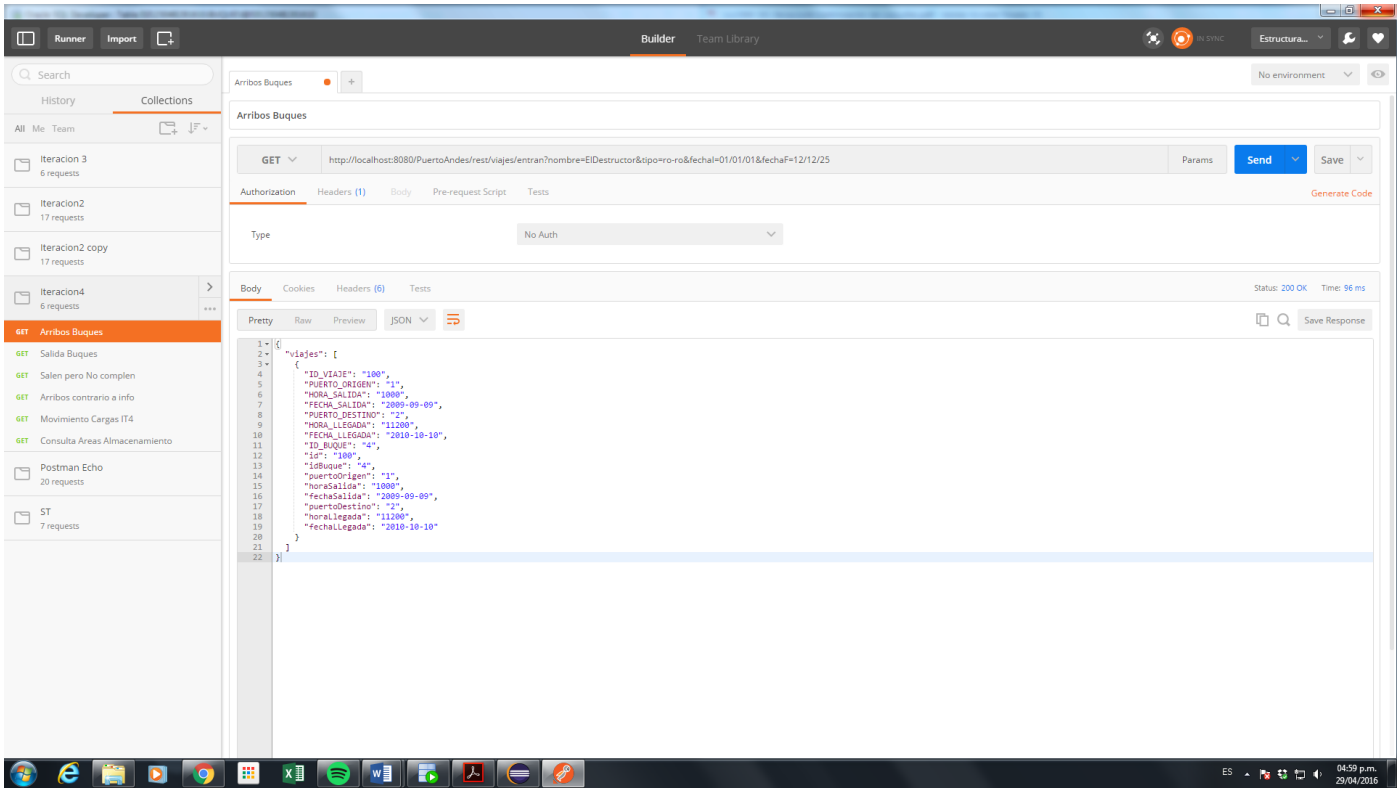
## 4. Plan de consulta

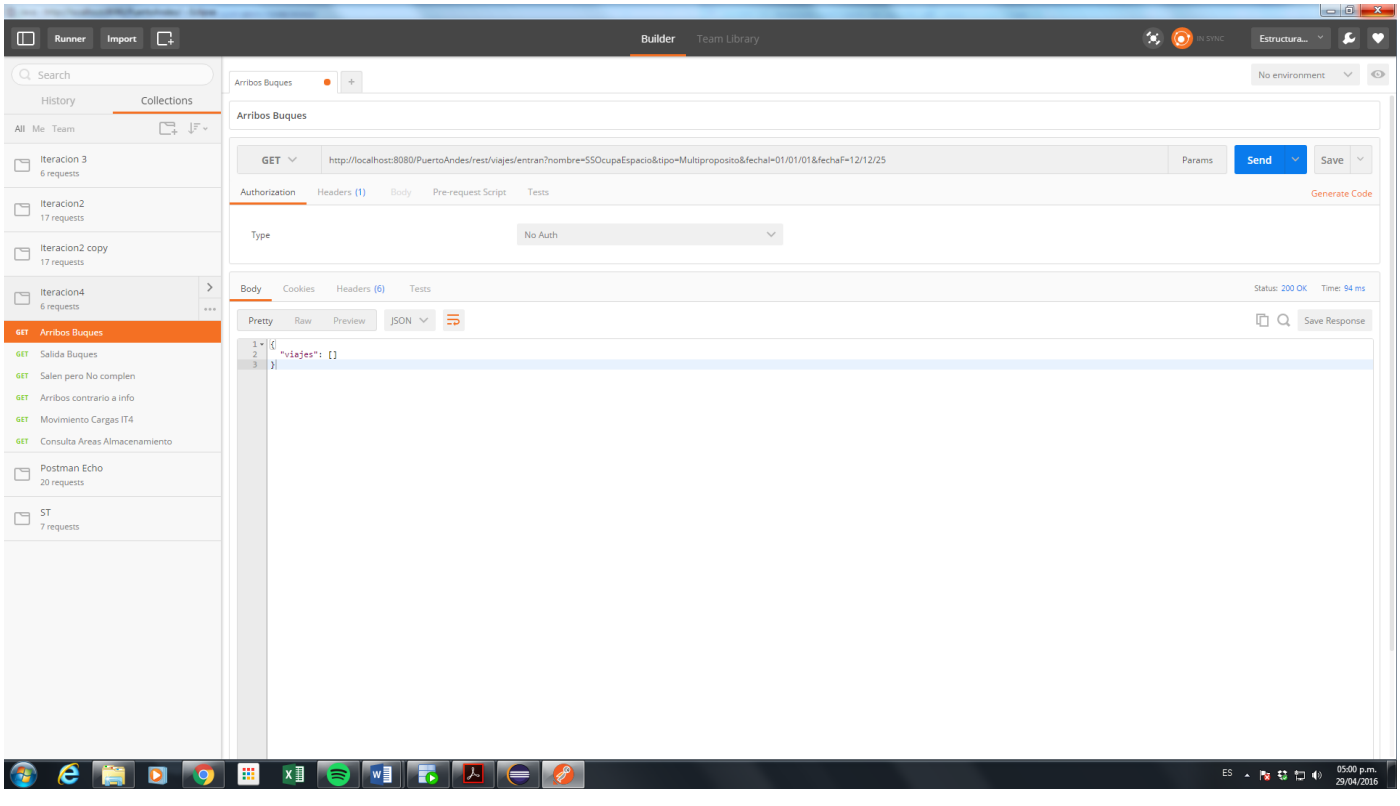



## 5. Tiempos Obtenidos



6. Escenarios



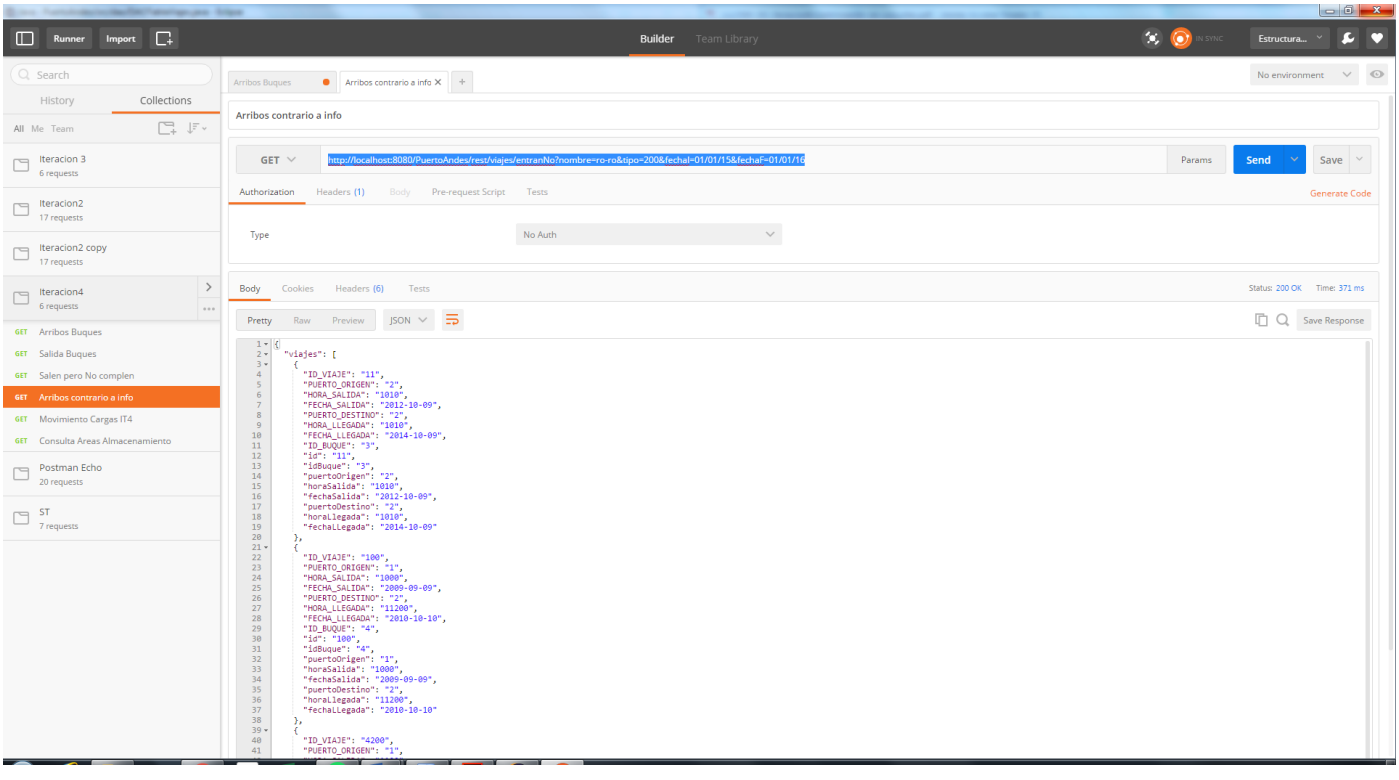
7. Plan de ejecución:  Fechas entre rango -> busca Buques que estén en viajes -> selecciona los buques que cumplan con el nombre y tipo que entra por parámetro.

RFC8. CONSULTAR LOS ARRIBOS Y SALIDAS DE BUQUES EN PUERTOANDES – RFC7-v2

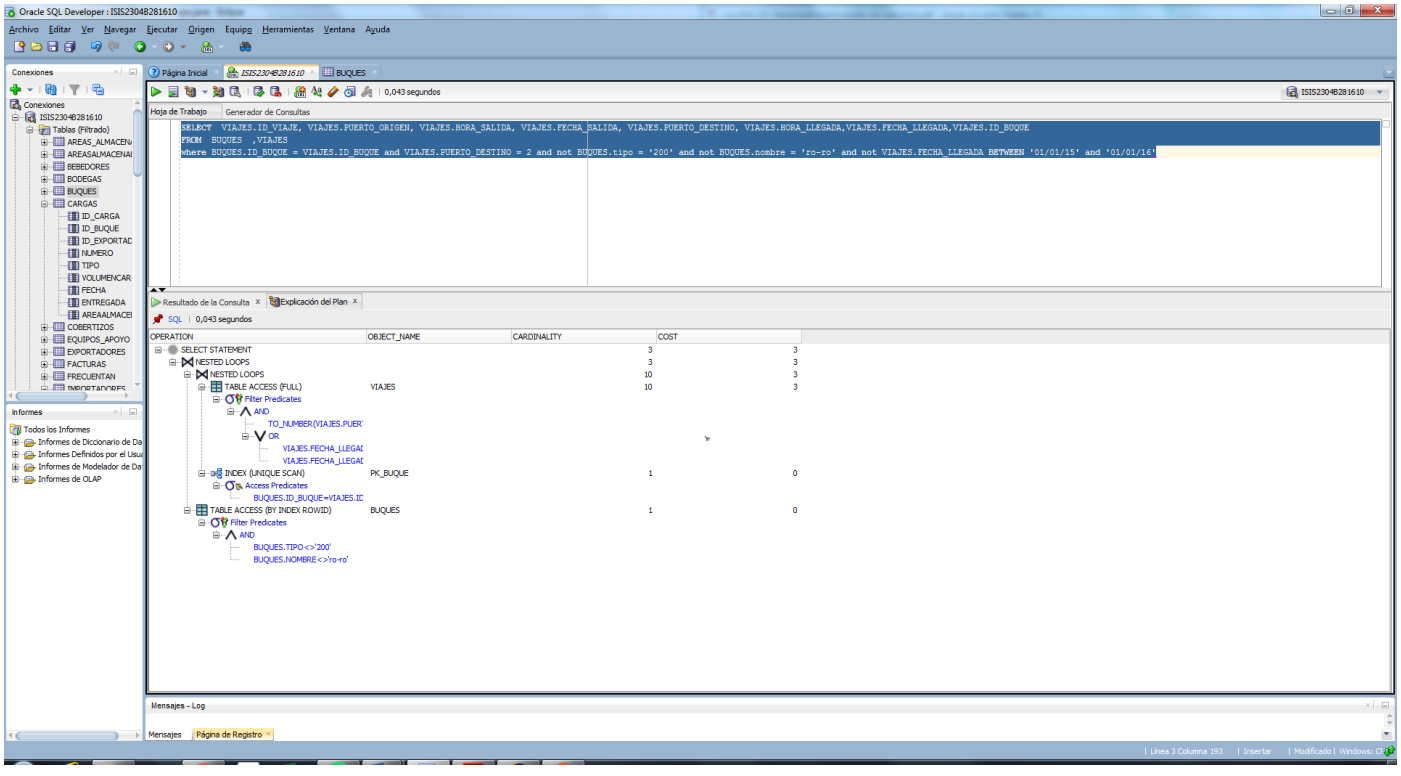
1. Clausula SQL utilizada:

"SELECT VIAJES.ID_VIAJE, VIAJES.PUERTO_ORIGEN, VIAJES.HORA_SALIDA, VIAJES.FECHA_SALIDA, VIAJES.PUERTO_DESTINO, VIAJES.HORA_LLEGADA,VIAJES.FECHA_LLEGADA,VIAJES.ID_BUQUE FROM BUQUES ,VIAJES where BUQUES.ID_BUQUE = VIAJES.ID_BUQUE and VIAJES.PUERTO_DESTINO = 2 and not BUQUES.tipo = '"+tipoBuque+"' and not BUQUES.nombre = '"+nombreBuque+"' and not VIAJES.FECHA_LLEGADA BETWEEN '"+fechaInicial+"' and '"+ fechaFinal+"'";

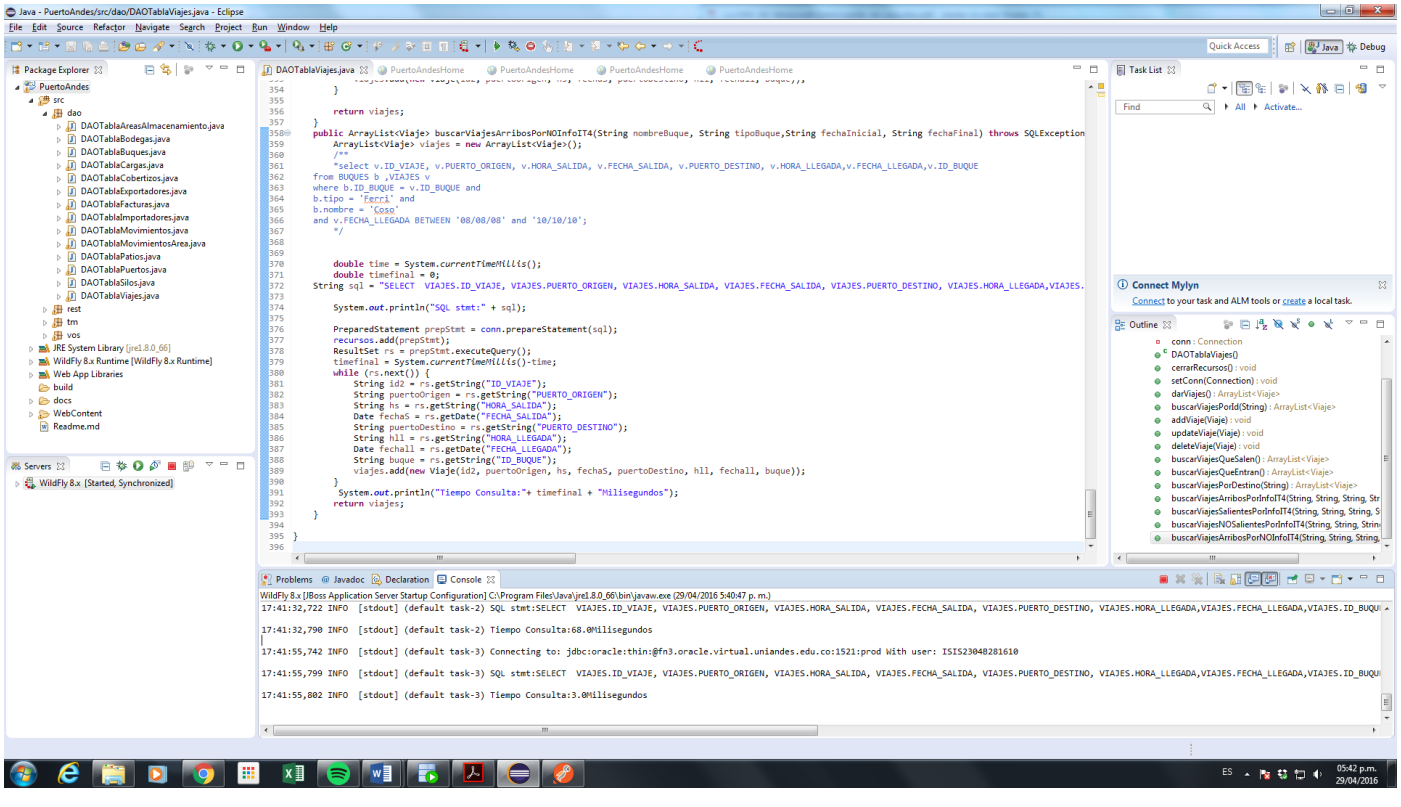
2. En este caso será todo lo contrario entre más tuplas hagan match con los parámetros ingresados menos serán las respuestas que se le muestren al usuario ya que en esta búsqueda se muestran las tuplas que no cumplen ninguna de los parámetros de búsqueda exceptuando los rangos de fechas
3. Valores Parametros en la prueba



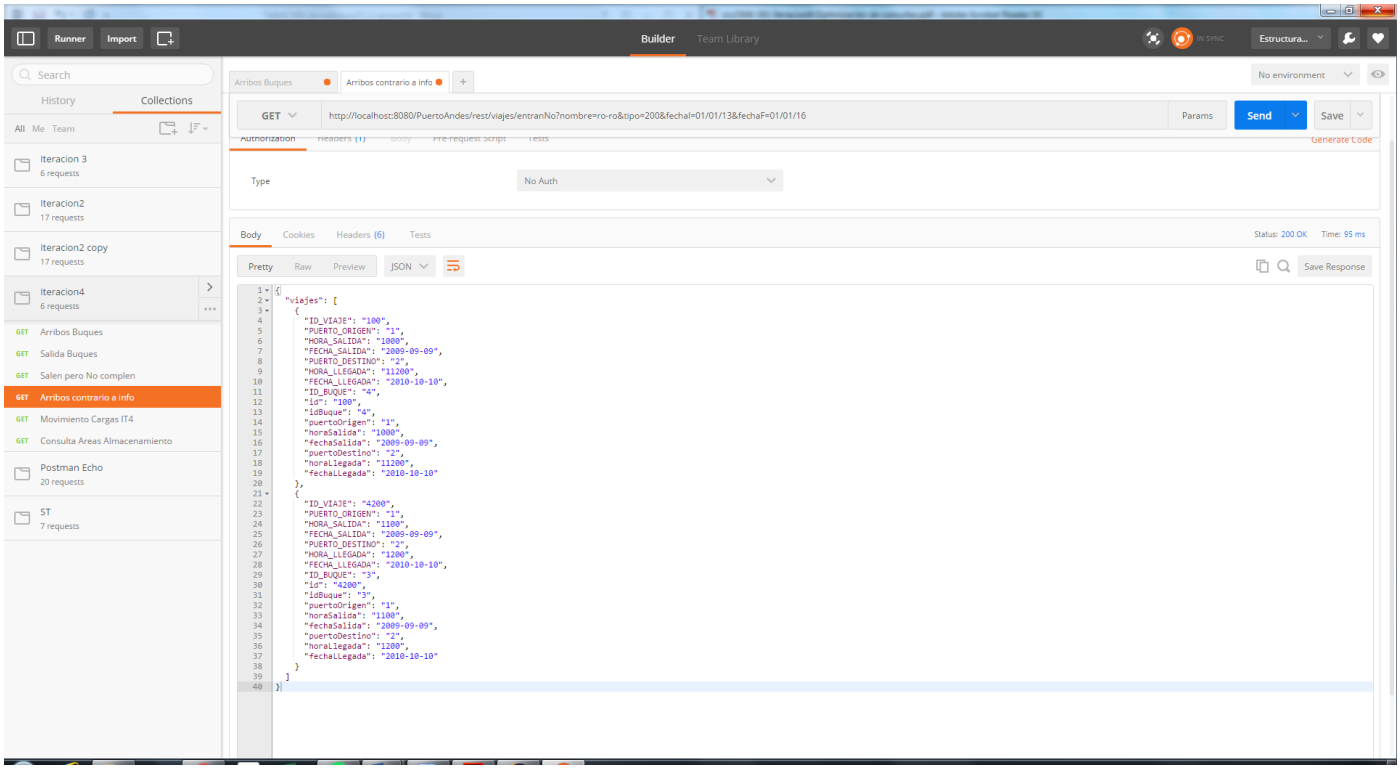
4.

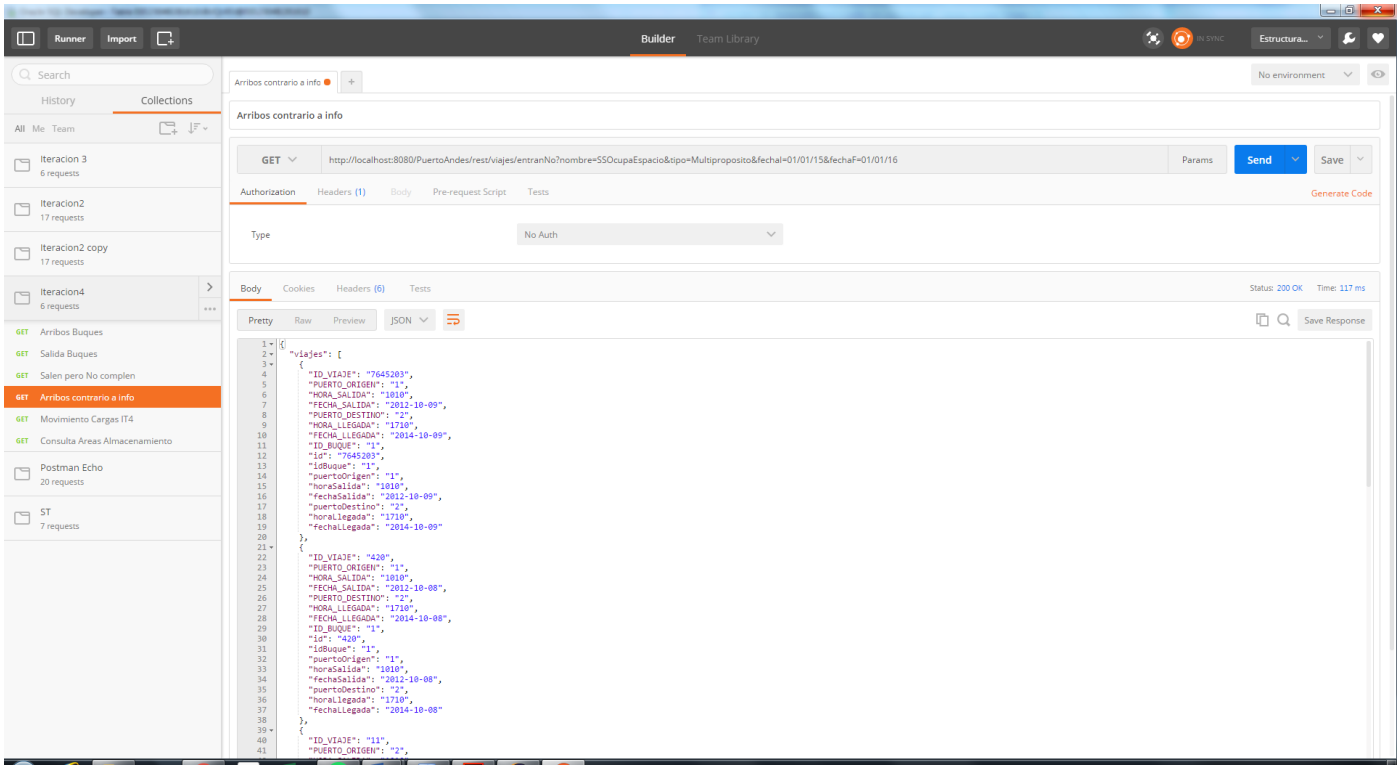

Oracle SQL Developer : ISIS2304B281610
SELECT VIAJES.ID_VIAJE, VIAJES.PUERTO_ORIGEN, VIAJES.HORA_SALIDA, VIAJES.FECHA_SALIDA, VIAJES.PUERTO_DESTINO, VIAJES.HORA_LLEGADA,VIAJES.FECHA_LLEGADA,VIAJES.ID_BUQUE
FROM BUQUES ,VIAJES
where BUQUES.ID_BUQUE = VIAJES.ID_BUQUE and VIAJES.PUERTO_DESTINO = 2 and not BUQUES.tipo = '200' and not BUQUES.nombre = 'ro-ro' and not VIAJES.FECHA_LLEGADA BETWEEN '01/01/15' and '01/01/16'
OPERATION
OBJECT_NAME
CARDINALITY
COST
SELECT STATEMENT
NESTED LOOPS
NESTED LOOPS
TABLE ACCESS (FULL)
VIAJES
Filter Predicates
AND
TO_NUMBER(VIAJES.PUER
OR
VIAJES.FECHA_LLEGAI
VIAJES.FECHA_LLEGAI
INDEX (UNIQUE SCAN)
PK_BUQUE
Access Predicates
BUQUES.ID_BUQUE=VIAJES.IC
TABLE ACCESS (BY INDEX ROWID)
BUQUES
Filter Predicates
AND
BUQUES.TIPO <>'200'
BUQUES.NOMBRE <>'ro-ro'

5.


Java - PuertoAndes/src/dao/DAOTablaViajes.java - Eclipse
DAOTablaViajes.java
public ArrayList<Viaje> buscarViajesArribosPorNOInfoIT4(String nombreBuque, String tipoBuque,String fechaInicial, String fechaFinal) throws SQLException
ArrayList<Viaje> viajes = new ArrayList<Viaje>();
"select v.ID_VIAJE, v.PUERTO_ORIGEN, v.HORA_SALIDA, v.FECHA_SALIDA, v.PUERTO_DESTINO, v.HORA_LLEGADA,v.FECHA_LLEGADA,v.ID_BUQUE
from BUQUES b ,VIAJES v
where b.ID_BUQUE = v.ID_BUQUE and
b.tipo = 'Ferri' and
b.nombre = 'Coso'
and v.FECHA_LLEGADA BETWEEN '08/08/08' and '10/10/10';
double time = System.currentTimeMillis();
double timefinal = 0;
String sql = "SELECT VIAJES.ID_VIAJE, VIAJES.PUERTO_ORIGEN, VIAJES.HORA_SALIDA, VIAJES.FECHA_SALIDA, VIAJES.PUERTO_DESTINO, VIAJES.HORA_LLEGADA,VIAJES.
System.out.println("SQL stmt:" + sql);
PreparedStatement prepStmt = conn.prepareStatement(sql);
recursos.add(prepStmt);
ResultSet rs = prepStmt.executeQuery();
timefinal = System.currentTimeMillis()-time;
while (rs.next()) {
String id2 = rs.getString("ID_VIAJE");
String puertoOrigen = rs.getString("PUERTO_ORIGEN");
String hs = rs.getString("HORA_SALIDA");
Date fechaS = rs.getDate("FECHA_SALIDA");
String puertoDestino = rs.getString("PUERTO_DESTINO");
String hll = rs.getString("HORA_LLEGADA");
Date fechall = rs.getDate("FECHA_LLEGADA");
String buque = rs.getString("ID_BUQUE");
viajes.add(new Viaje(id2, puertoOrigen, hs, fechaS, puertoDestino, hll, fechall, buque));
System.out.println("Tiempo Consulta:"+ timefinal + "Milisegundos");
return viajes;
WildFly 8.x [JBoss Application Server Startup Configuration] C:\Program Files\Java\jre1.8.0_66\bin\javaw.exe (29/04/2016 5:40:47 p. m.)
17:41:32,722 INFO [stdout] (default task-2) SQL stmt:SELECT VIAJES.ID_VIAJE, VIAJES.PUERTO_ORIGEN, VIAJES.HORA_SALIDA, VIAJES.FECHA_SALIDA, VIAJES.PUERTO_DESTINO, VIAJES.HORA_LLEGADA,VIAJES.FECHA_LLEGADA,VIAJES.ID_BUQUE
17:41:32,790 INFO [stdout] (default task-2) Tiempo Consulta:68.0Milisegundos
17:41:55,742 INFO [stdout] (default task-3) Connecting to: jdbc:oracle:thin:@fn3.oracle.virtual.uniandes.edu.co:1521:prod With user: ISIS2304B281610
17:41:55,799 INFO [stdout] (default task-3) SQL stmt:SELECT VIAJES.ID_VIAJE, VIAJES.PUERTO_ORIGEN, VIAJES.HORA_SALIDA, VIAJES.FECHA_SALIDA, VIAJES.PUERTO_DESTINO, VIAJES.HORA_LLEGADA,VIAJES.FECHA_LLEGADA,VIAJES.ID_BUQUE
17:41:55,802 INFO [stdout] (default task-3) Tiempo Consulta:3.0Milisegundos

6.


Runner
Import
Builder
Team Library
Search
History
Collections
Iteracion 3
6 requests
Iteracion2
17 requests
Iteracion2 copy
17 requests
Iteracion4
6 requests
GET Arribos Buques
GET Salida Buques
GET Salen pero No complen
GET Arribos contrario a info
GET Movimiento Cargas IT4
GET Consulta Areas Almacenamiento
Postman Echo
20 requests
ST
7 requests
Arribos Buques
Arribos contrario a info
No environment
GET
http://localhost:8080/PuertoAndes/rest/viajes/entranNo?nombre=ro-ro&tipo=200&fechaI=01/01/13&fechaF=01/01/16
Params
Send
Save
Authorization
Headers (1)
Body
Pre-request Script
Tests
Generate Code
Type
No Auth
Body
Cookies
Headers (6)
Tests
Status: 200 OK
Time: 95 ms
Pretty
Raw
Preview
JSON
Save Response
{
"viajes": [
{
"ID_VIAJE": "100",
"PUERTO_ORIGEN": "1",
"HORA_SALIDA": "1000",
"FECHA_SALIDA": "2009-09-09",
"PUERTO_DESTINO": "2",
"HORA_LLEGADA": "11200",
"FECHA_LLEGADA": "2010-10-10",
"ID_BUQUE": "4",
"id": "100",
"idBuque": "4",
"puertoOrigen": "1",
"horaSalida": "1000",
"fechaSalida": "2009-09-09",
"puertoDestino": "2",
"horaLlegada": "11200",
"fechaLlegada": "2010-10-10"
},
{
"ID_VIAJE": "4200",
"PUERTO_ORIGEN": "1",
"HORA_SALIDA": "1100",
"FECHA_SALIDA": "2009-09-09",
"PUERTO_DESTINO": "2",
"HORA_LLEGADA": "1200",
"FECHA_LLEGADA": "2010-10-10",
"ID_BUQUE": "3",
"id": "4200",
"idBuque": "3",
"puertoOrigen": "1",
"horaSalida": "1100",
"fechaSalida": "2009-09-09",
"puertoDestino": "2",
"horaLlegada": "1200",
"fechaLlegada": "2010-10-10"
}
]
}

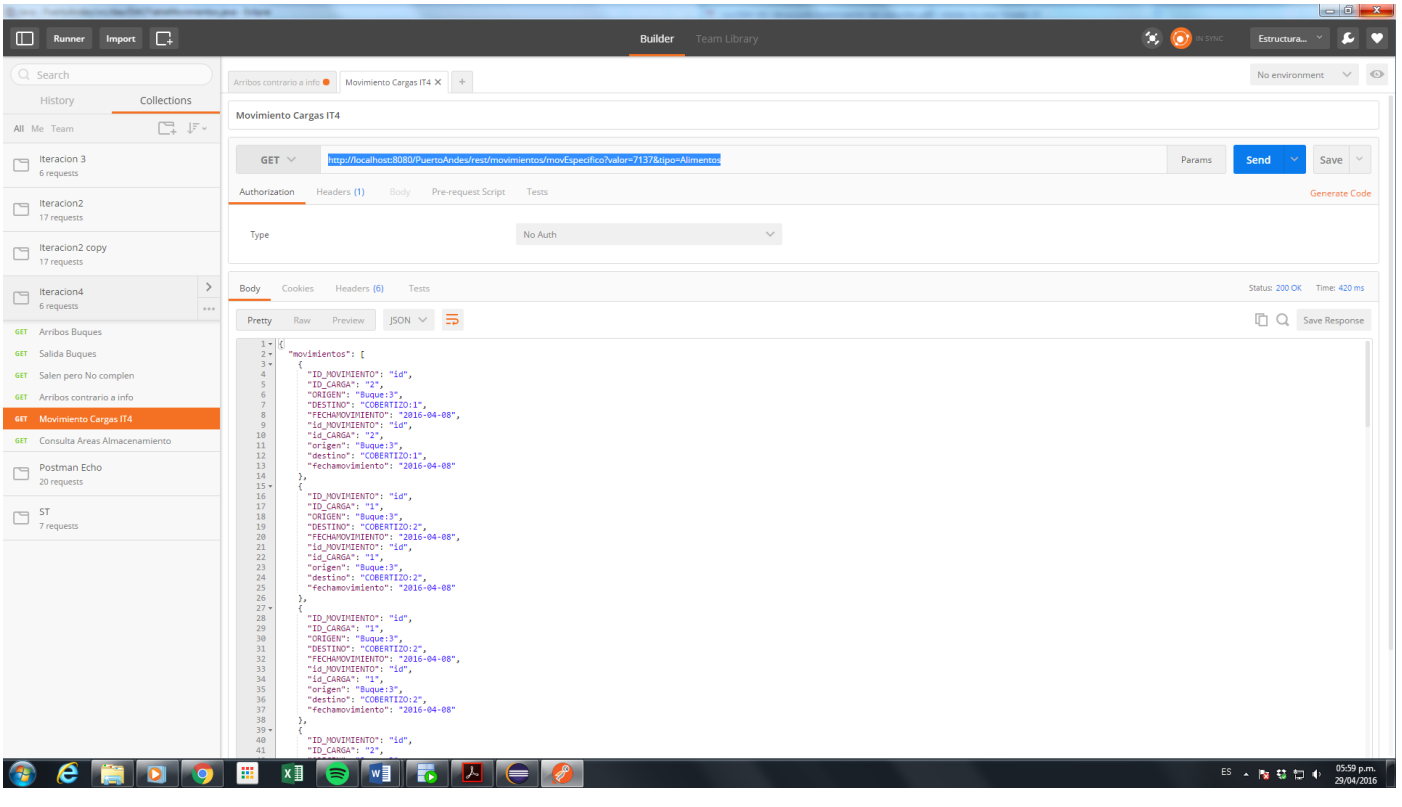

Runner
Import
Builder
Team Library
Search
History
Collections
Iteracion 3
6 requests
Iteracion2
17 requests
Iteracion2 copy
17 requests
Iteracion4
6 requests
GET Arribos Buques
GET Salida Buques
GET Salen pero No complen
GET Arribos contrario a info
GET Movimiento Cargas IT4
GET Consulta Areas Almacenamiento
Postman Echo
20 requests
ST
7 requests
Arribos contrario a info
No environment
Arribos contrario a info
GET
http://localhost:8080/PuertoAndes/rest/viajes/entranNo?nombre=SSOcupaEspacio&tipo=Multiproposito&fechaI=01/01/15&fechaF=01/01/16
Params
Send
Save
Authorization
Headers (1)
Body
Pre-request Script
Tests
Generate Code
Type
No Auth
Body
Cookies
Headers (6)
Tests
Status: 200 OK
Time: 117 ms
Pretty
Raw
Preview
JSON
Save Response
{
"viajes": [
{
"ID_VIAJE": "7645203",
"PUERTO_ORIGEN": "1",
"HORA_SALIDA": "1010",
"FECHA_SALIDA": "2012-10-09",
"PUERTO_DESTINO": "2",
"HORA_LLEGADA": "1710",
"FECHA_LLEGADA": "2014-10-09",
"ID_BUQUE": "1",
"id": "7645203",
"idBuque": "1",
"puertoOrigen": "1",
"horaSalida": "1010",
"fechaSalida": "2012-10-09",
"puertoDestino": "2",
"horaLlegada": "1710",
"fechaLlegada": "2014-10-09"
},
{
"ID_VIAJE": "420",
"PUERTO_ORIGEN": "1",
"HORA_SALIDA": "1010",
"FECHA_SALIDA": "2012-10-08",
"PUERTO_DESTINO": "2",
"HORA_LLEGADA": "1710",
"FECHA_LLEGADA": "2014-10-08",
"ID_BUQUE": "1",
"id": "420",
"idBuque": "1",
"puertoOrigen": "1",
"horaSalida": "1010",
"fechaSalida": "2012-10-08",
"puertoDestino": "2",
"horaLlegada": "1710",
"fechaLlegada": "2014-10-08"
},
{
"ID_VIAJE": "11",
"PUERTO_ORIGEN": "2",

7. Plan de ejecución:  Fechas entre rango -> busca Buques que estén en viajes -> selecciona los buques que sean diferentes al nombre y tipo que entran por parámetro.

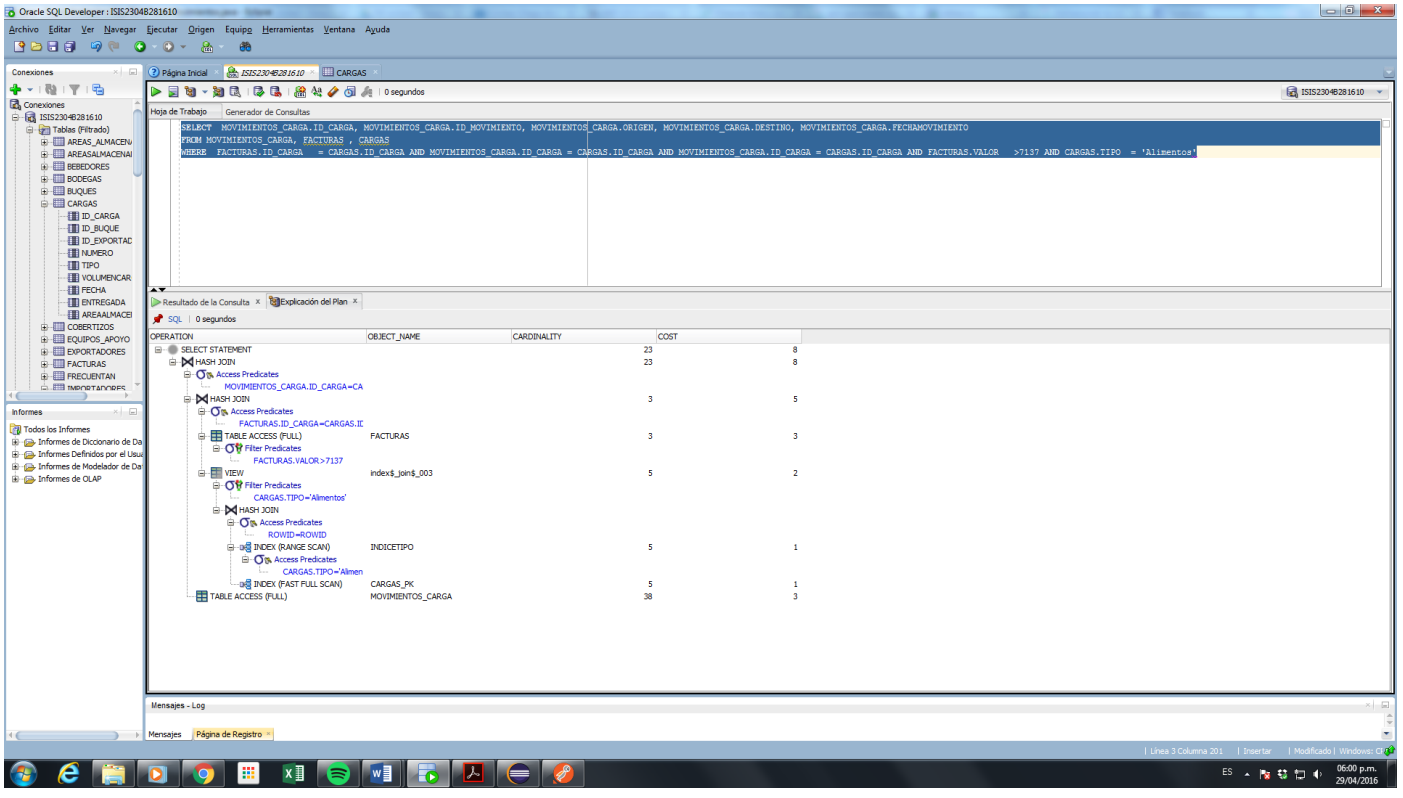
RFC9. CONSULTAR MOVIMIENTOS DE CARGA – RFC5-v2

1. Sentencia SQL seleccionada:

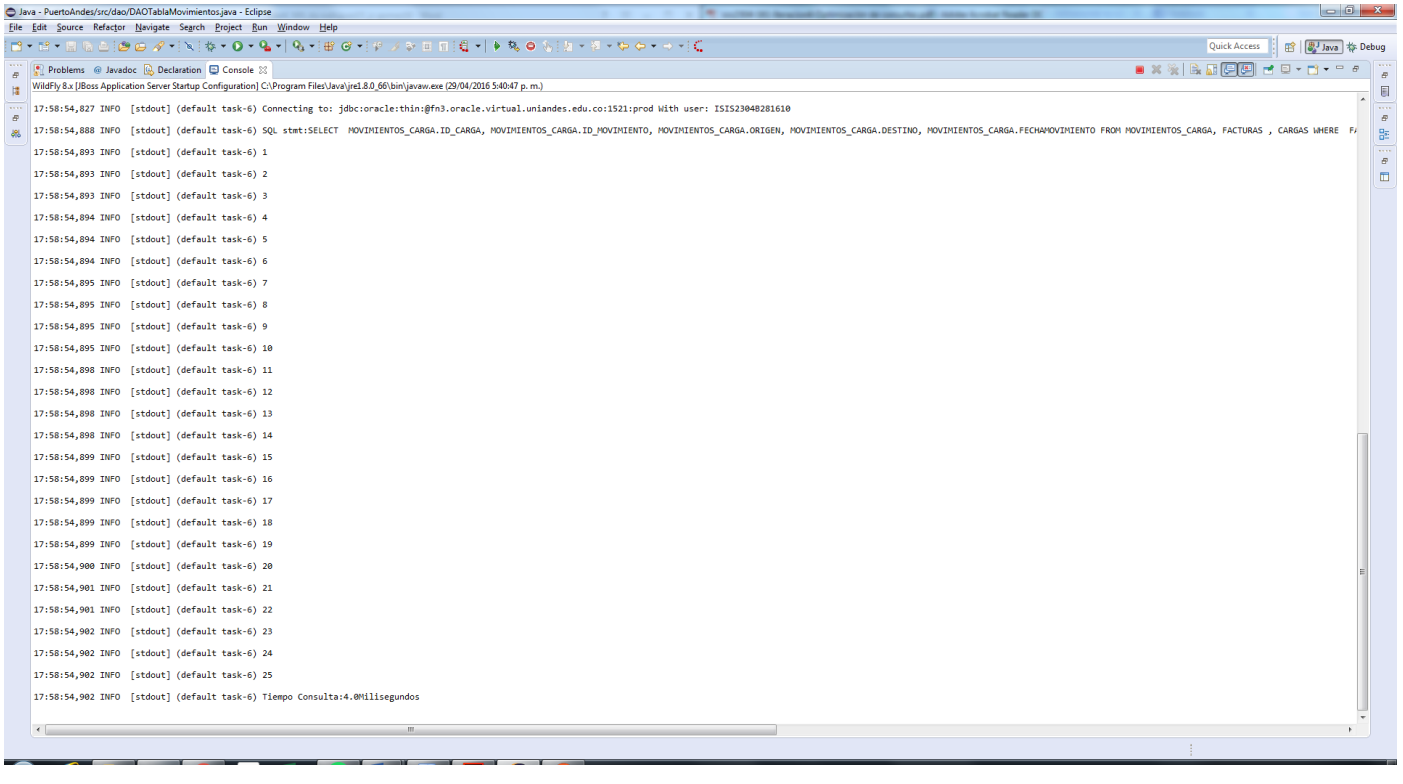
"SELECT MOVIMIENTOS_CARGA.ID_CARGA, MOVIMIENTOS_CARGA.ID_MOVIMIENTO, MOVIMIENTOS_CARGA.ORIGEN, MOVIMIENTOS_CARGA.DESTINO, MOVIMIENTOS_CARGA.FECHAMOVIMIENTO FROM MOVIMIENTOS_CARGA, FACTURAS , CARGAS WHERE FACTURAS.ID_CARGA = CARGAS.ID_CARGA AND MOVIMIENTOS_CARGA.ID_CARGA = CARGAS.ID_CARGA AND MOVIMIENTOS_CARGA.ID_CARGA = CARGAS.ID_CARGA AND FACTURAS.VALOR >"+valor +" AND CARGAS.TIPO = "'+ tipoCarga+"'";

2. El tamaño varía según el número de facturas y al valor mínimo de la factura, ya que buscamos los movimientos de carga asociados a las cargas que tengan facturado un valor mayor a X y que cumplan con el tipo de carga que entra por parámetro, el tamaño de la respuesta depende del tipo de carga, ya que en ocasiones hay más cargas de un tipo que de otro.
3. 

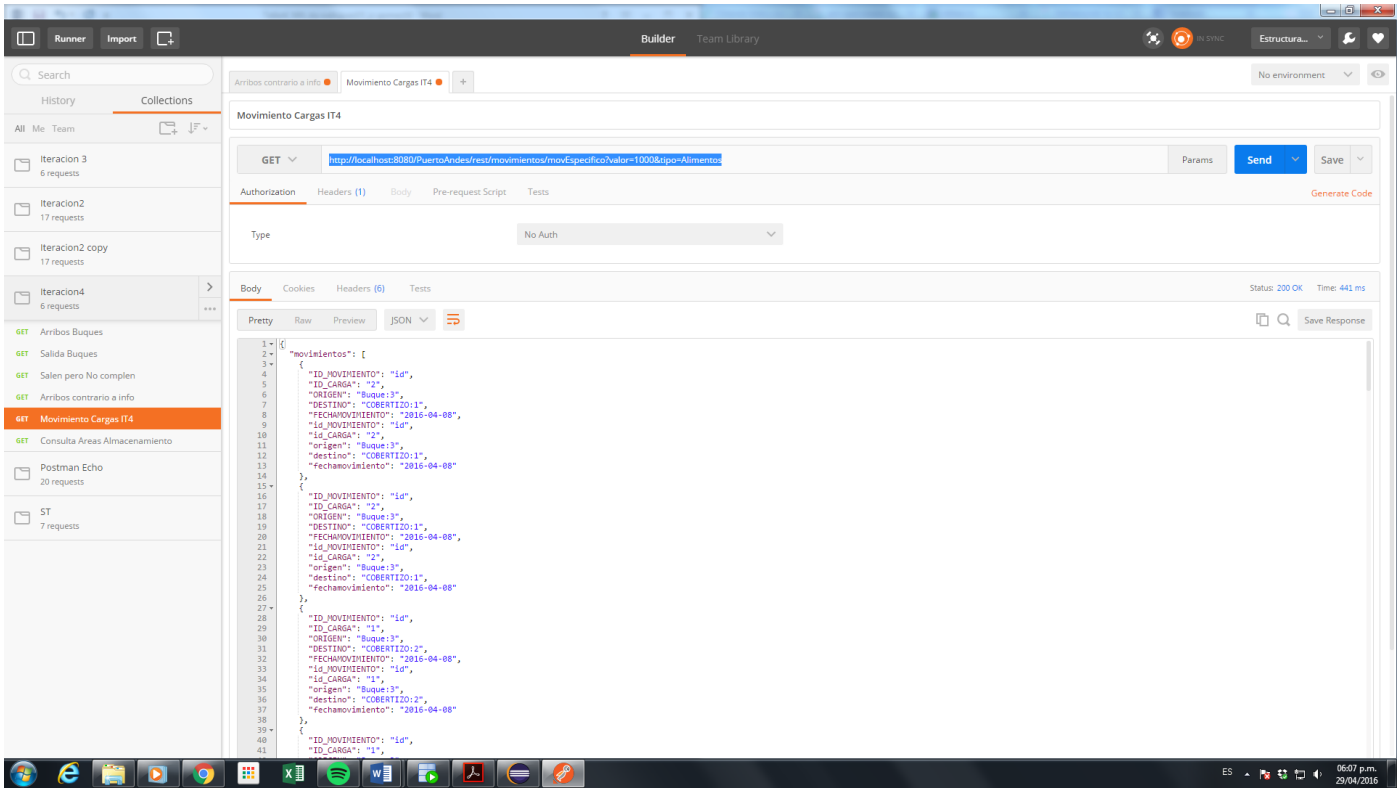
4.

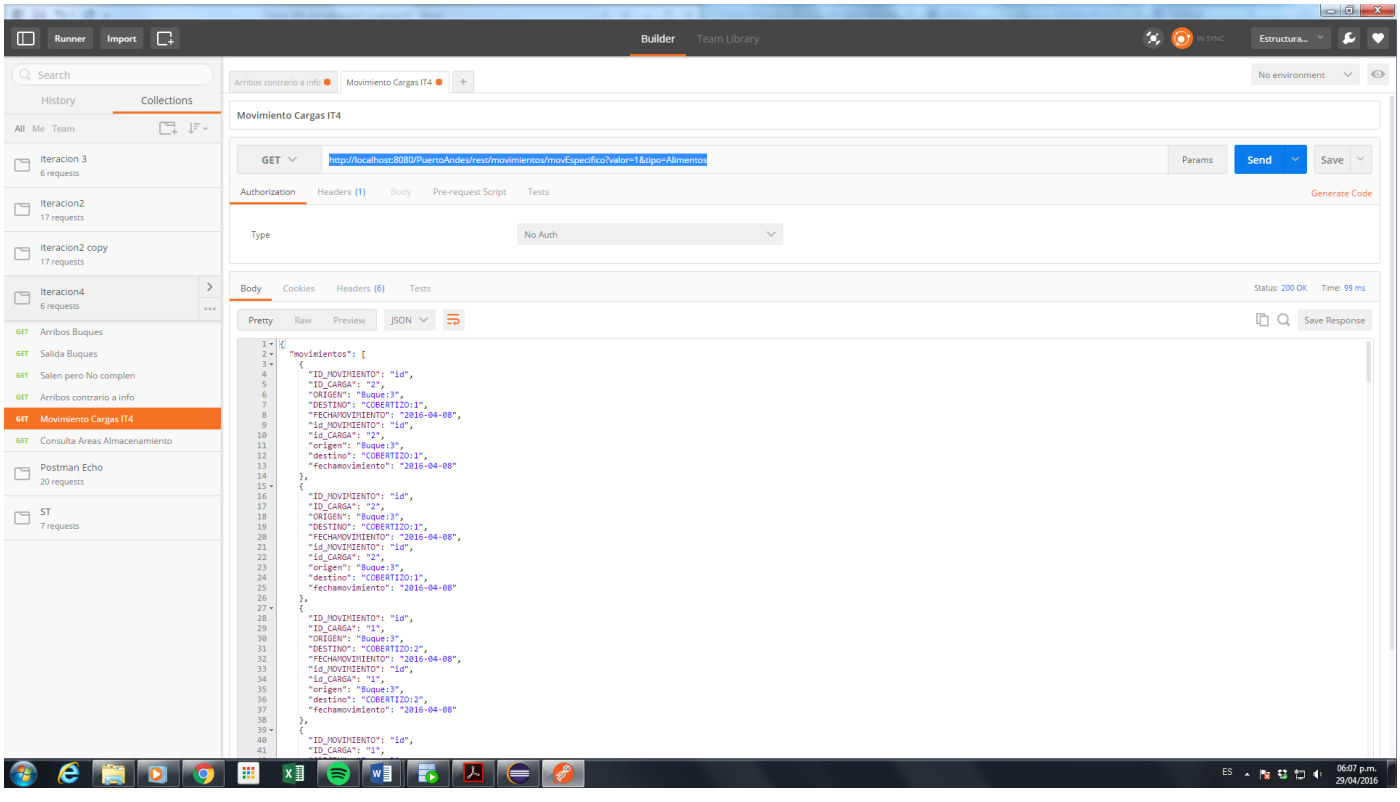

Oracle SQL Developer : ISIS2304B281610
SELECT MOVIMIENTOS_CARGA.ID_CARGA, MOVIMIENTOS_CARGA.ID_MOVIMIENTO, MOVIMIENTOS_CARGA.ORIGEN, MOVIMIENTOS_CARGA.DESTINO, MOVIMIENTOS_CARGA.FECHAMOVIMIENTO
FROM MOVIMIENTOS_CARGA, FACTURAS , CARGAS
WHERE FACTURAS.ID_CARGA = CARGAS.ID_CARGA AND MOVIMIENTOS_CARGA.ID_CARGA = CARGAS.ID_CARGA AND MOVIMIENTOS_CARGA.ID_CARGA = CARGAS.ID_CARGA AND FACTURAS.VALOR >7137 AND CARGAS.TIPO = 'Alimentos'
OPERATION OBJECT_NAME CARDINALITY COST
SELECT STATEMENT 23 8
HASH JOIN 23 8
Access Predicates
MOVIMIENTOS_CARGA.ID_CARGA=CA
HASH JOIN 3 5
Access Predicates
FACTURAS.ID_CARGA=CARGAS.II
TABLE ACCESS (FULL) FACTURAS 3 3
Filter Predicates
FACTURAS.VALOR>7137
VIEW index$_join$_003 5 2
Filter Predicates
CARGAS.TIPO='Alimentos'
HASH JOIN
Access Predicates
ROWID=ROWID
INDEX (RANGE SCAN) INDICETIPO 5 1
Access Predicates
CARGAS.TIPO='Almen
INDEX (FAST FULL SCAN) CARGAS_PK 5 1
TABLE ACCESS (FULL) MOVIMIENTOS_CARGA 38 3

5.


Java - PuertoAndes/src/dao/DAOTablaMovimientos.java - Eclipse
17:58:54,827 INFO [stdout] (default task-6) Connecting to: jdbc:oracle:thin:@fn3.oracle.virtual.uniandes.edu.co:1521:prod With user: ISIS2304B281610
17:58:54,888 INFO [stdout] (default task-6) SQL stmt:SELECT MOVIMIENTOS_CARGA.ID_CARGA, MOVIMIENTOS_CARGA.ID_MOVIMIENTO, MOVIMIENTOS_CARGA.ORIGEN, MOVIMIENTOS_CARGA.DESTINO, MOVIMIENTOS_CARGA.FECHAMOVIMIENTO FROM MOVIMIENTOS_CARGA, FACTURAS , CARGAS WHERE
17:58:54,893 INFO [stdout] (default task-6) 1
17:58:54,893 INFO [stdout] (default task-6) 2
17:58:54,893 INFO [stdout] (default task-6) 3
17:58:54,894 INFO [stdout] (default task-6) 4
17:58:54,894 INFO [stdout] (default task-6) 5
17:58:54,894 INFO [stdout] (default task-6) 6
17:58:54,895 INFO [stdout] (default task-6) 7
17:58:54,895 INFO [stdout] (default task-6) 8
17:58:54,895 INFO [stdout] (default task-6) 9
17:58:54,895 INFO [stdout] (default task-6) 10
17:58:54,898 INFO [stdout] (default task-6) 11
17:58:54,898 INFO [stdout] (default task-6) 12
17:58:54,898 INFO [stdout] (default task-6) 13
17:58:54,898 INFO [stdout] (default task-6) 14
17:58:54,899 INFO [stdout] (default task-6) 15
17:58:54,899 INFO [stdout] (default task-6) 16
17:58:54,899 INFO [stdout] (default task-6) 17
17:58:54,899 INFO [stdout] (default task-6) 18
17:58:54,899 INFO [stdout] (default task-6) 19
17:58:54,900 INFO [stdout] (default task-6) 20
17:58:54,901 INFO [stdout] (default task-6) 21
17:58:54,901 INFO [stdout] (default task-6) 22
17:58:54,902 INFO [stdout] (default task-6) 23
17:58:54,902 INFO [stdout] (default task-6) 24
17:58:54,902 INFO [stdout] (default task-6) 25
17:58:54,902 INFO [stdout] (default task-6) Tiempo Consulta:4.0Milisegundos

6.


Runner
Import
Builder
Team Library
Search
History
Collections
Iteracion 3
6 requests
Iteracion2
17 requests
Iteracion2 copy
17 requests
Iteracion4
6 requests
GET Arribos Buques
GET Salida Buques
GET Salen pero No complen
GET Arribos contrario a info
GET Movimiento Cargas IT4
GET Consulta Areas Almacenamiento
Postman Echo
20 requests
ST
7 requests
Movimiento Cargas IT4
GET
http://localhost:8080/PuertoAndes/rest/movimientos/movEspecifico?valor=1000&tipo=Alimentos
Params
Send
Save
Authorization
Headers (1)
Pre-request Script
Tests
Generate Code
Type
No Auth
Body
Cookies
Headers (6)
Tests
Status: 200 OK
Time: 441 ms
Pretty
Raw
Preview
JSON
Save Response
{
"movimientos": [
{
"ID_MOVIMIENTO": "id",
"ID_CARGA": "2",
"ORIGEN": "Buque:3",
"DESTINO": "COBERTIZO:1",
"FECHAMOVIMIENTO": "2016-04-08",
"id_MOVIMIENTO": "id",
"id_CARGA": "2",
"origen": "Buque:3",
"destino": "COBERTIZO:1",
"fechamovimiento": "2016-04-08"
},
{
"ID_MOVIMIENTO": "id",
"ID_CARGA": "2",
"ORIGEN": "Buque:3",
"DESTINO": "COBERTIZO:1",
"FECHAMOVIMIENTO": "2016-04-08",
"id_MOVIMIENTO": "id",
"id_CARGA": "2",
"origen": "Buque:3",
"destino": "COBERTIZO:1",
"fechamovimiento": "2016-04-08"
},
{
"ID_MOVIMIENTO": "id",
"ID_CARGA": "1",
"ORIGEN": "Buque:3",
"DESTINO": "COBERTIZO:2",
"FECHAMOVIMIENTO": "2016-04-08",
"id_MOVIMIENTO": "id",
"id_CARGA": "1",
"origen": "Buque:3",
"destino": "COBERTIZO:2",
"fechamovimiento": "2016-04-08"
},
{
"ID_MOVIMIENTO": "id",
"ID_CARGA": "1",
06:07 p.m.
29/04/2016

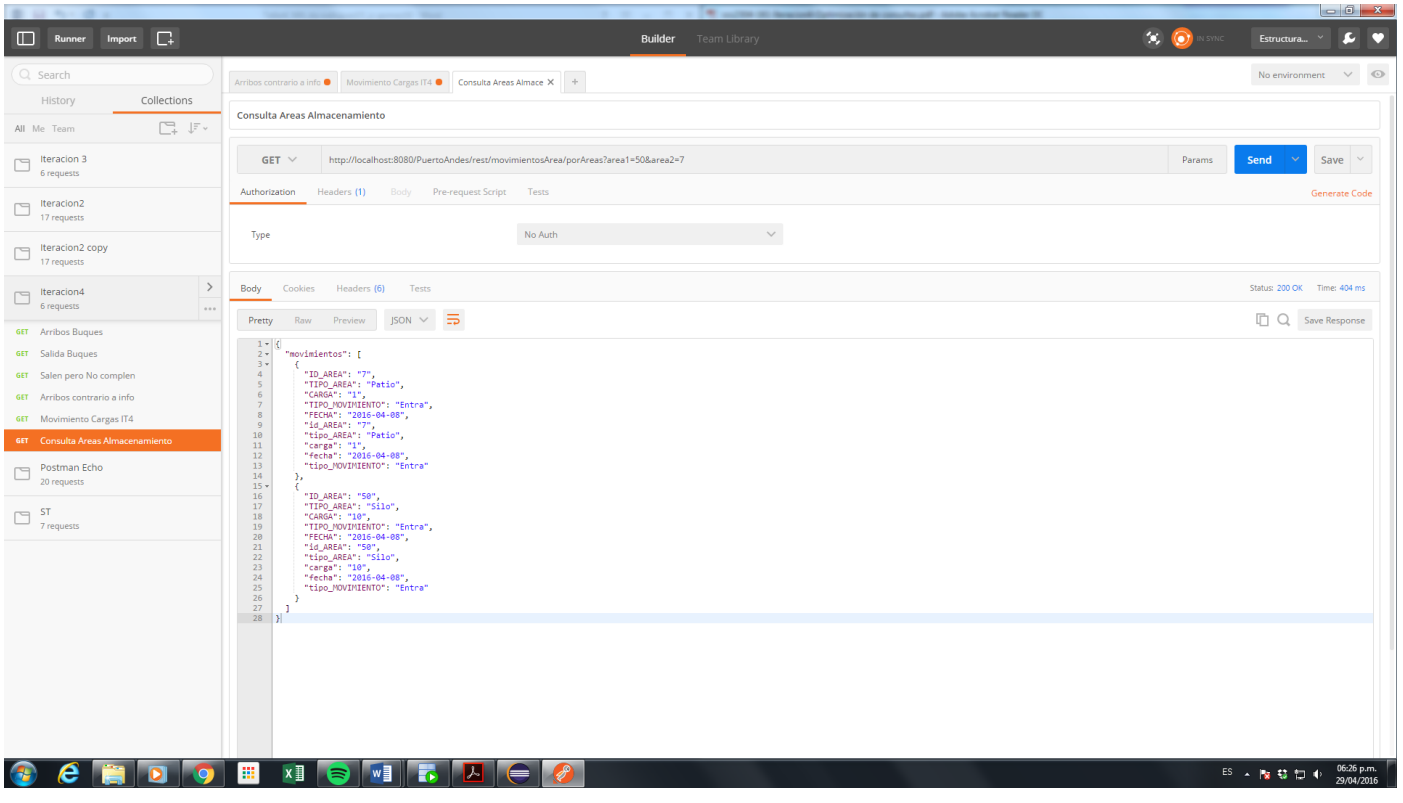

Runner
Import
Builder
Team Library
Search
History
Collections
Iteracion 3
6 requests
Iteracion2
17 requests
Iteracion2 copy
17 requests
Iteracion4
6 requests
GET Arribos Buques
GET Salida Buques
GET Salen pero No complen
GET Arribos contrario a info
GET Movimiento Cargas IT4
GET Consulta Areas Almacenamiento
Postman Echo
20 requests
ST
7 requests
Movimiento Cargas IT4
GET
http://localhost:8080/PuertoAndes/rest/movimientos/movEspecifico?valor=1&tipo=Alimentos
Params
Send
Save
Authorization
Headers (1)
Pre-request Script
Tests
Generate Code
Type
No Auth
Body
Cookies
Headers (6)
Tests
Status: 200 OK
Time: 99 ms
Pretty
Raw
Preview
JSON
Save Response
{
"movimientos": [
{
"ID_MOVIMIENTO": "id",
"ID_CARGA": "2",
"ORIGEN": "Buque:3",
"DESTINO": "COBERTIZO:1",
"FECHAMOVIMIENTO": "2016-04-08",
"id_MOVIMIENTO": "id",
"id_CARGA": "2",
"origen": "Buque:3",
"destino": "COBERTIZO:1",
"fechamovimiento": "2016-04-08"
},
{
"ID_MOVIMIENTO": "id",
"ID_CARGA": "2",
"ORIGEN": "Buque:3",
"DESTINO": "COBERTIZO:1",
"FECHAMOVIMIENTO": "2016-04-08",
"id_MOVIMIENTO": "id",
"id_CARGA": "2",
"origen": "Buque:3",
"destino": "COBERTIZO:1",
"fechamovimiento": "2016-04-08"
},
{
"ID_MOVIMIENTO": "id",
"ID_CARGA": "1",
"ORIGEN": "Buque:3",
"DESTINO": "COBERTIZO:2",
"FECHAMOVIMIENTO": "2016-04-08",
"id_MOVIMIENTO": "id",
"id_CARGA": "1",
"origen": "Buque:3",
"destino": "COBERTIZO:2",
"fechamovimiento": "2016-04-08"
},
{
"ID_MOVIMIENTO": "id",
"ID_CARGA": "1",
06:07 p.m.
29/04/2016

7 Plan de ejecución: buscamos las cargas que están presentes en proceso de carga -> buscamos las cargas que están en facturas -> de esas cargas seleccionamos las que las facturas sean mayores a X -> Comparamos que el tipo de carga sea alimentos -> Respuesta

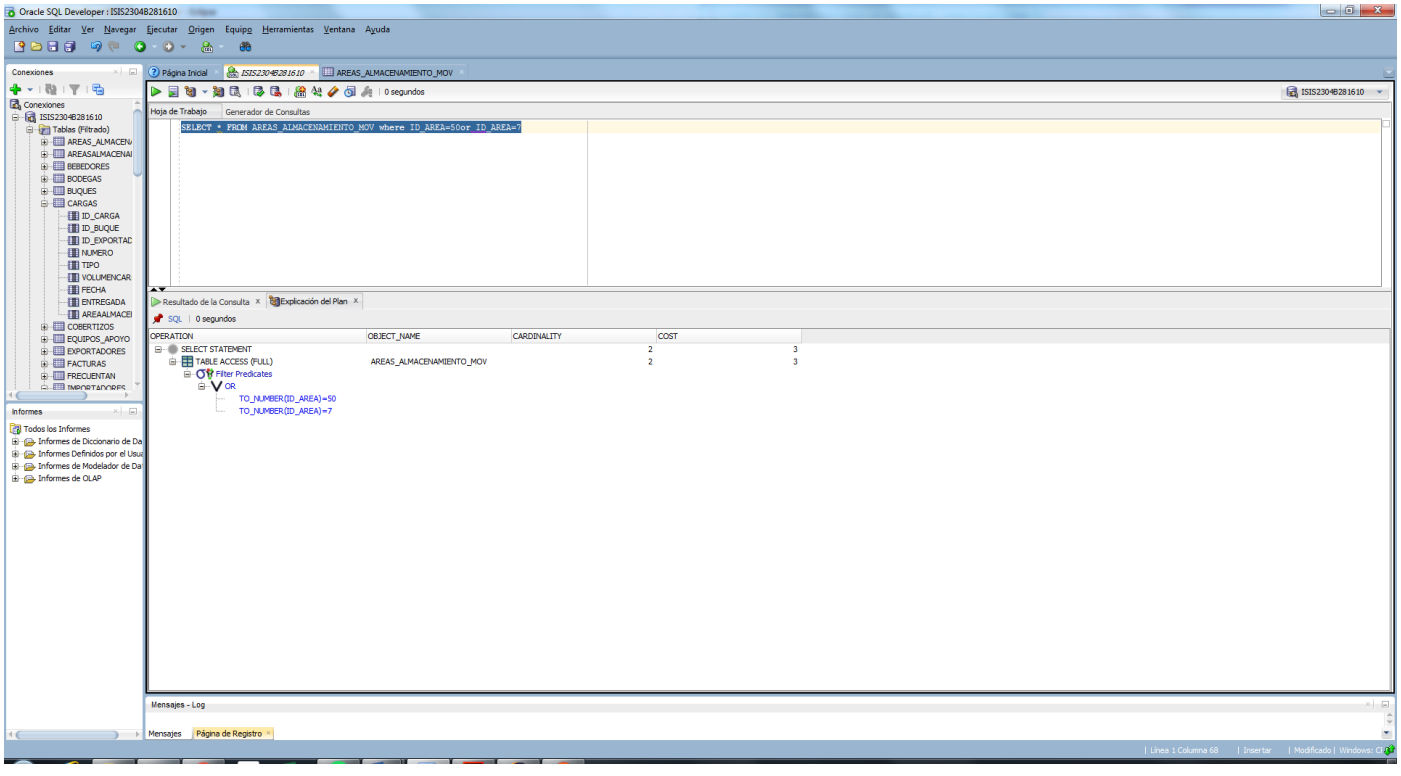
RFC10. CONSULTAR ÁREAS DE ALMACENAMIENTO – RFC6-v2

1. Sentencia SQL utilizada:

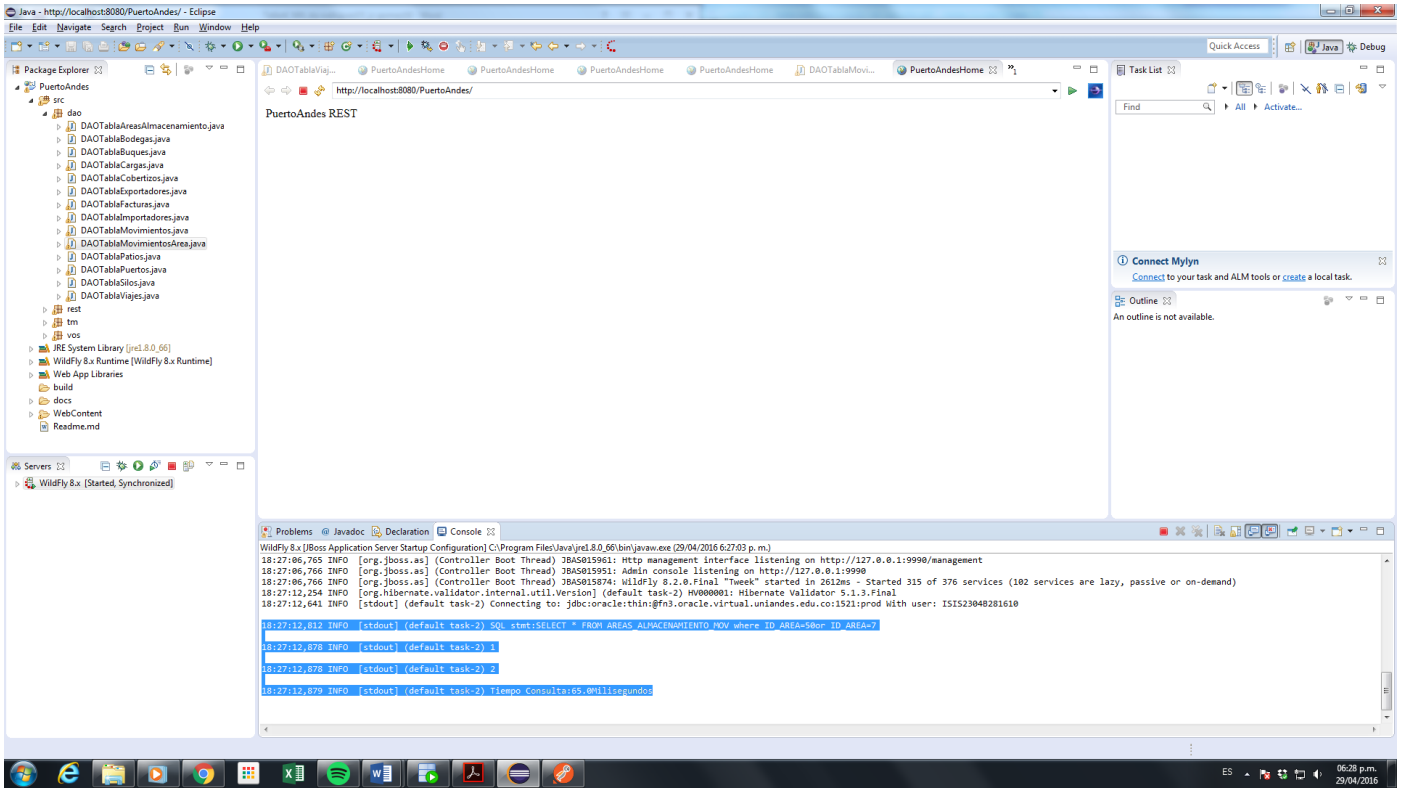
SELECT * FROM AREAS_ALMACENAMIENTO_MOV where ID_AREA="+idArea1+"or ID_AREA="+idArea2;

2. El tamaño de la respuesta varia según la afluencia de movimientos que hayan en esta área de almacenamiento, por ejemplo si el área es muy usada habrán muchas respuestas, el tamaño de la respuesta es difícil e saber debido a que el programa asigna de manera automática las áreas de almacenamiento
3. 

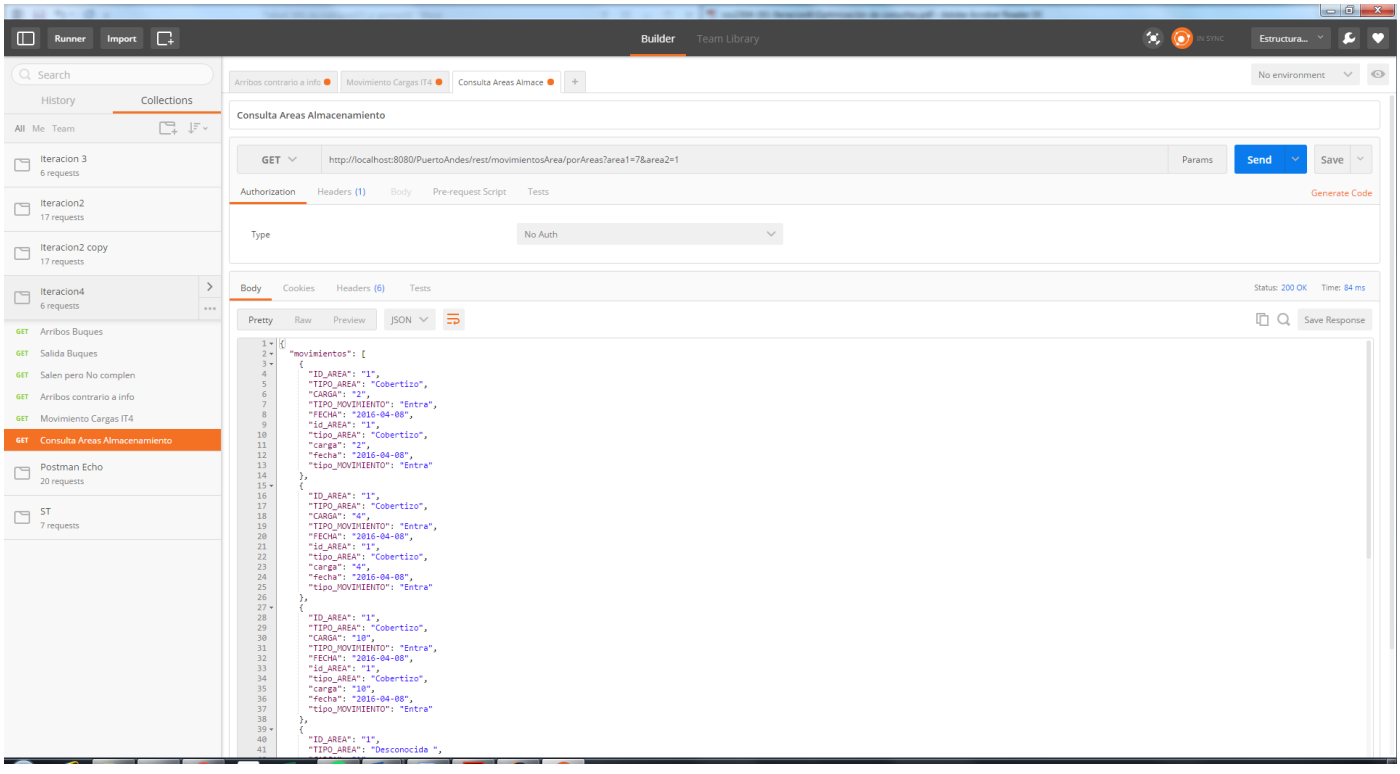
4.

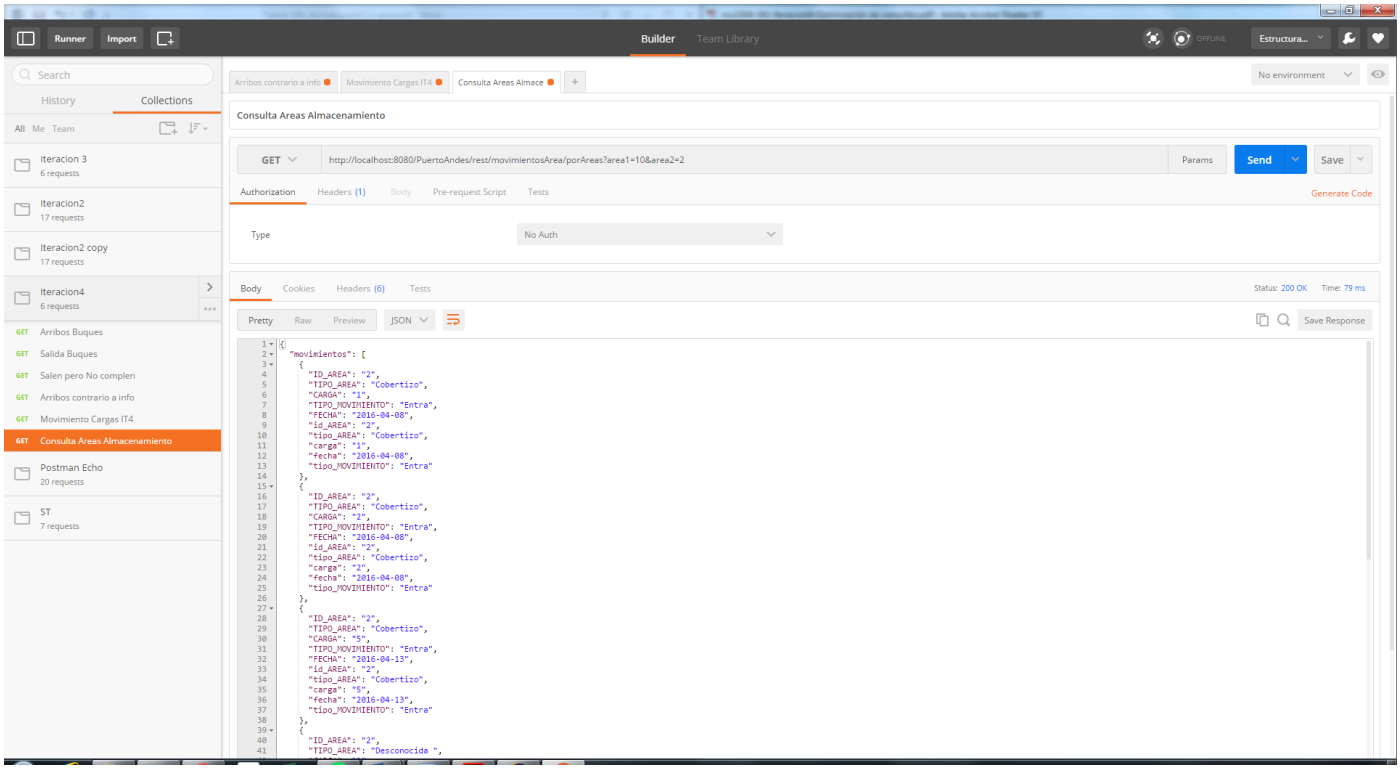

Oracle SQL Developer : ISIS2304B281610
SELECT * FROM AREAS_ALMACENAMIENTO_MOV where ID_AREA=50or ID_AREA=7
OPERATION
OBJECT_NAME
CARDINALITY
COST
SELECT STATEMENT
TABLE ACCESS (FULL)
AREAS_ALMACENAMIENTO_MOV
Filter Predicates
OR
TO_NUMBER(ID_AREA)=50
TO_NUMBER(ID_AREA)=7

5.


Java - http://localhost:8080/PuertoAndes/ - Eclipse
PuertoAndes REST
18:27:06,765 INFO [org.jboss.as] (Controller Boot Thread) JBAS015961: Http management interface listening on http://127.0.0.1:9990/management
18:27:06,766 INFO [org.jboss.as] (Controller Boot Thread) JBAS015951: Admin console listening on http://127.0.0.1:9990
18:27:06,766 INFO [org.jboss.as] (Controller Boot Thread) JBAS015874: WildFly 8.2.0.Final "Tweek" started in 2612ms - Started 315 of 376 services (102 services are lazy, passive or on-demand)
18:27:12,254 INFO [org.hibernate.validator.internal.util.Version] (default task-2) HV000001: Hibernate Validator 5.1.3.Final
18:27:12,641 INFO [stdout] (default task-2) Connecting to: jdbc:oracle:thin:@fn3.oracle.virtual.uniandes.edu.co:1521:prod With user: ISIS2304B281610
18:27:12,812 INFO [stdout] (default task-2) SQL stmt:SELECT * FROM AREAS_ALMACENAMIENTO_MOV where ID_AREA=50or ID_AREA=7
18:27:12,878 INFO [stdout] (default task-2) 1
18:27:12,878 INFO [stdout] (default task-2) 2
18:27:12,879 INFO [stdout] (default task-2) Tiempo Consulta:65.0Milisegundos

6.

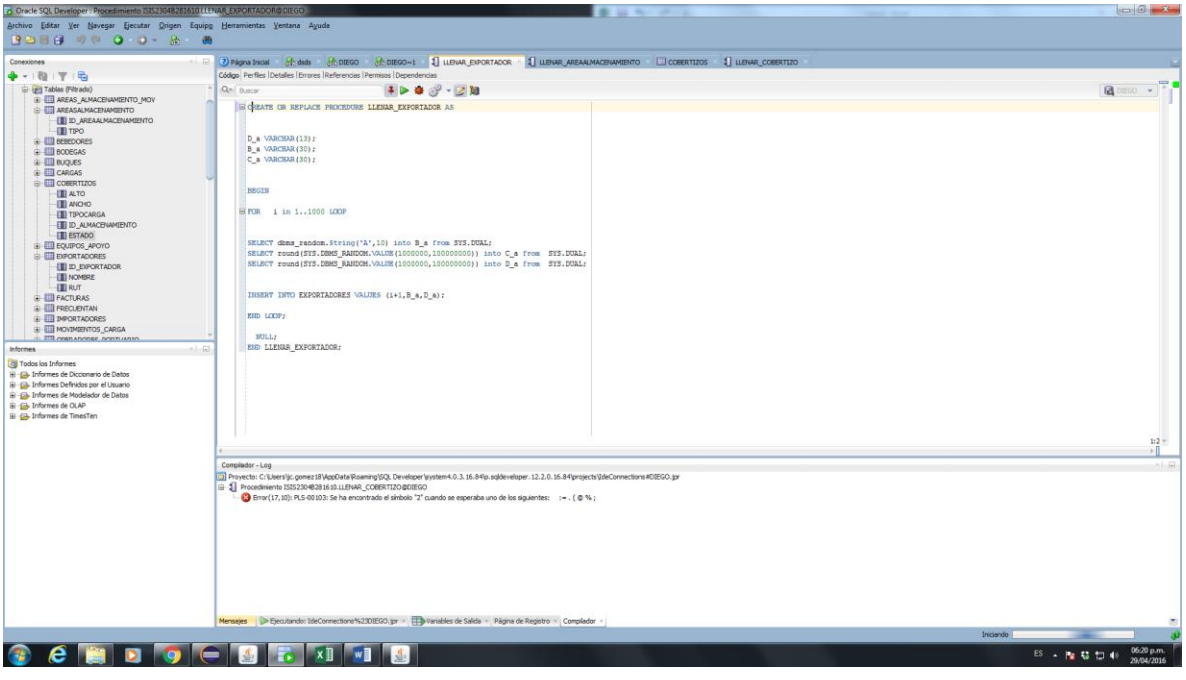

Runner
Import
Builder
Team Library
Consulta Areas Almacenamiento
GET
http://localhost:8080/PuertoAndes/rest/movimientosArea/porAreas?area1=7&area2=1
Send
Save
Authorization
Headers (1)
Pre-request Script
Tests
Generate Code
No Auth
Body
Cookies
Headers (6)
Status: 200 OK
Time: 84 ms
Pretty
Raw
Preview
JSON
Save Response
{
"movimientos": [
{
"ID_AREA": "1",
"TIPO_AREA": "Cobertizo",
"CARGA": "2",
"TIPO_MOVIMIENTO": "Entra",
"FECHA": "2016-04-08",
"id_AREA": "1",
"tipo_AREA": "Cobertizo",
"carga": "2",
"fecha": "2016-04-08",
"tipo_MOVIMIENTO": "Entra"
},
{
"ID_AREA": "1",
"TIPO_AREA": "Cobertizo",
"CARGA": "4",
"TIPO_MOVIMIENTO": "Entra",
"FECHA": "2016-04-08",
"id_AREA": "1",
"tipo_AREA": "Cobertizo",
"carga": "4",
"fecha": "2016-04-08",
"tipo_MOVIMIENTO": "Entra"
},
{
"ID_AREA": "1",
"TIPO_AREA": "Cobertizo",
"CARGA": "10",
"TIPO_MOVIMIENTO": "Entra",
"FECHA": "2016-04-08",
"id_AREA": "1",
"tipo_AREA": "Cobertizo",
"carga": "10",
"fecha": "2016-04-08",
"tipo_MOVIMIENTO": "Entra"
},
{
"ID_AREA": "1",
"TIPO_AREA": "Desconocida ",


Runner
Import
Builder
Team Library
Consulta Areas Almacenamiento
GET
http://localhost:8080/PuertoAndes/rest/movimientosArea/porAreas?area1=10&area2=2
Send
Save
Authorization
Headers (1)
Pre-request Script
Tests
Generate Code
No Auth
Body
Cookies
Headers (6)
Status: 200 OK
Time: 79 ms
Pretty
Raw
Preview
JSON
Save Response
{
"movimientos": [
{
"ID_AREA": "2",
"TIPO_AREA": "Cobertizo",
"CARGA": "1",
"TIPO_MOVIMIENTO": "Entra",
"FECHA": "2016-04-08",
"id_AREA": "2",
"tipo_AREA": "Cobertizo",
"carga": "1",
"fecha": "2016-04-08",
"tipo_MOVIMIENTO": "Entra"
},
{
"ID_AREA": "2",
"TIPO_AREA": "Cobertizo",
"CARGA": "2",
"TIPO_MOVIMIENTO": "Entra",
"FECHA": "2016-04-08",
"id_AREA": "2",
"tipo_AREA": "Cobertizo",
"carga": "2",
"fecha": "2016-04-08",
"tipo_MOVIMIENTO": "Entra"
},
{
"ID_AREA": "2",
"TIPO_AREA": "Cobertizo",
"CARGA": "5",
"TIPO_MOVIMIENTO": "Entra",
"FECHA": "2016-04-13",
"id_AREA": "2",
"tipo_AREA": "Cobertizo",
"carga": "5",
"fecha": "2016-04-13",
"tipo_MOVIMIENTO": "Entra"
},
{
"ID_AREA": "2",
"TIPO_AREA": "Desconocida ",

7. Plan de ejecución: Movimientos en las áreas que entran por parámetro.

**5%)** Documente claramente el proceso de carga de datos: Cómo fue realizado, cómo logró el volumen de datos solicitado.

Para probar la eficiencia de nuestras consuetas es necesario llenar las tablas con un volumen de datos lo suficientemente grandes, para este proceso usamos las herramientas disponibles en SQL Developer. Para cargar cada tabla fue necesario crear un procedimiento como el siguiente:




Pero como en las columnas de las diferentes tablas los datos son diferentes se modifica el tipo de valor que queremos que se genere aleatoriamente, por ejemplo si queremos un valor aleatorio de Strings se pone la siguiente sentencia: SELECT dbms_random.String('A',10) into B_a from SYS.DUAL; pero si por el contrario se requiere un numero en un rango se pone la siguiente sentencia: SELECT round(SYS.DBMS_RANDOM.VALUE(a,b)) into C_a from SYS.DUAL siendo a y b los valores numéricos entre los cuales queremos que se encuentre nuestro numero aleatorio.

(15 %) Análisis del proceso de optimización y el modelo de ejecución de consultas.

- Analice la diferencia entre la ejecución de consultas delegada al manejador de bases de datos como Oracle y compárelo con una ejecución donde la aplicación trae los datos a memoria principal y resuelve con instrucciones de control (if, while, etc.), los operadores involucrados en las consultas como joins, selecciones y proyecciones.

  En todo sentido es mucho mejor delegar las consultas a la base de datos, primero por el espacio en memoria: a considerar una gran carga de datos como por ejemplo 2,000,000 de datos y crear objetos en java con referencia a una entidad genera un costo muy grande de memoria, problema que seguramente va a desencadenar un java.lang.OutOfMemoryError, en caso de que no sea así y cree los 2,000,000 de objetos presentaríamos otro problema de rendimiento en nuestra solución, el problema es una demora de tiempo que se verá reflejado en el momento de dar la respuesta ya que el solo hecho de crear los datos a partir de la consulta SQL (toma más o menos un minuto o minuto y medio) , además de esto, debemos comparar objeto a objeto, para saber cuáles cumplen con los parámetros de búsqueda, para guardarlos en una lista y poder mostrárselos al usuario. Mientras que si le delegamos la búsqueda a la base de datos tendremos una mejora en cuento a memoria y a tiempos de ejecución, debido a que podemos utilizar los recursos de optimización de consultas que nos ofrezca la base de datos (en este caso Oracle) , como lo son los índices que usados de la forma adecuada nos permitirán reducir el tiempo y la complejidad en las consultas, una vez que hemos reducido el problema de tiempos de respuesta, debemos señalar el impacto que genera en la memoria; hacemos una sentencia SQL en donde realicemos la consulta deseada, obteniendo como respuesta las tuplas que cumplen con los criterios de búsqueda, generando solo los objetos que cumplan con la búsqueda, metiéndolos en una lista y presentándoselos al usuario; de esta manera no hacemos un uso innecesario de memoria ya que solo vamos a crear los objetos necesarios para dar respuesta a la consulta del usuario. Por estas razones es mejor la ejecución de consultas delegadas al manejador de bases de datos.