**Iteración 4**

Contenido

[1. Análisis 1](#_Toc481870492)

[2. Diseño de la aplicación 1](#_Toc481870493)

[2.1 Análisis de Impacto 1](#_Toc481870494)

[2.2 Diseño Físico 1](#_Toc481870495)

[2.2.1 Justificación sobre los índices 1](#_Toc481870496)

[2.2.2 Diseño de Datos 3](#_Toc481870497)

[2.2.3 Documentación 4](#_Toc481870498)

# **Análisis**

Para el desarrollo de esta iteración no se realizan cambios sobre el modelo conceptual.

# **Diseño de la aplicación**

## **Análisis de Impacto**

Los nuevos requerimientos no implican un cambio sustancial del modelo conceptual, puesto que ya se tiene la información necesaria para operar sobre los requerimientos. Sin embargo, sobre el modelo relacional se efectúan cambios sobre los índices que se utilizan en las tablas. Estos índices hacen de forma eficiente el acceso a la información asociada a estos. Los índices se implementan bajo estructuras de datos distintas con el fin de garantizar un acceso óptimo a la información sin que esto consuma demasiada memoria principal. Como mejora de la iteración pasada, se implementan correctamente los niveles de aislamiento necesarios en los diferentes requerimientos.

## **Diseño Físico**

### **Justificación sobre los índices**

Las elecciones sobre los índices se realizaron basados en las operaciones más costosas sobre el plan de ejecución de Oracle. Así, tratando de garantizar un acceso rápido y adecuado a la información solicitada garantizando una lectura correcta sobre estos. Estos índices hacen de las consultas más eficientes y dado que no es una gran cantidad de índices, no afectan dramáticamente la memoria principal y su mantenimiento no es costoso.

#### **RFC9**

* + **Función-Fecha:** Este índice facilita el acceso de las fechas de las funciones, criterio por el cual se realiza la consulta del requerimiento. Este índice se asume como un árbol B+, dado que las fechas tendrían un puntero hacía la siguiente fecha, permitiendo acceder a un rango de fechas o a una fecha en específico de forma ágil y eficiente. Este índice abarca parcialmente la llave primaria de la función, sin embargo, no es el único criterio de agrupamiento, siendo así, es un índice secundario.
  + **Boleta-Función:** Uno de los *Join* más frecuentes entre los requerimientos es el *Join* entre Boleta y Función, por lo tanto, es un índice que es necesario para los requerimientos del negocio. Este índice igualmente se toma como un árbol B+, dado que puede haber varias boletas en una función, descartando la posibilidad de Hash, y un Bitmap no sería eficiente. Siendo el árbol B+ la mejor opción para esta implementación, además, las boletas de la función quedarían contiguas. Los atributos de la función alojados en la tabla Boletas, no definen la ubicación de los archivos ni su secuencialidad, por lo tanto, es un índice secundario.

#### **RFC10**

Los índices sobre esta tabla están estrechamente ligados con el requerimiento funcional 9, sin verse afectados en su totalidad.

#### **RFC11**

* + **Localidad-Nombre:** Para este requerimiento, dado que se expresa la necesidad de realizar un filtro por localidad, resulta práctico el índice de la localidad sobre el nombre de esta. En este caso, la implementación de este índice se realiza sobre una tabla de Hash, dado que dos Localidades no deberían tener el mismo nombre, siendo así, resulta práctico el acceso de estos en el menor tiempo posible. Además, al tener este Hash los *Join* al hacerlos por este criterio resultan más eficientes por medio de *Hash Join*. El nombre de la localidad es un atributo que no es llave candidata, por lo tanto, no es llave primaria de este, haciendo de este índice uno secundario.
  + **Boletas-Usuario:** En este caso, es de gran utilidad conocer la información completa del usuario quien es propietario de la boleta. Así, este *Join* se puede realizar eficientemente. En este caso, dado que un Usuario puede tener más de una Boleta, por lo tanto, se utiliza la estructura de árbol B+ para el almacenamiento de los índices. El usuario no hace parte de la llave primaria de la tabla Boletas, por lo tanto, no define la ubicación de las tuplas de esta haciéndolo un índice secundario.
  + **Función-Fecha:** Igualmente es necesaria la búsqueda por rangos de fechas, por lo que aquí también se ve reflejado el uso del índice, con su implementación de árbol B+. Este índice abarca parcialmente la llave primaria de la función, sin embargo, no es el único criterio de agrupamiento, siendo así, es un índice secundario.
  + **Requerimiento-Nombre:** Para el desarrollo del requerimiento, es necesario buscar por nombre del requerimiento técnico (Objetos del escenario), por lo tanto, la implementación de un índice, facilita el acceso a estos en menor tiempo y facilita el *Join* con esta tabla. Este índice, se implementa bajo la estructura de Hash dado que es un valor único para la tabla. El nombre del requerimiento es un atributo que no es llave candidata, por lo tanto, no es llave primaria de este, haciendo de este índice uno secundario.

#### **RFC12**

* + **Boleta-Usuario:** En este requerimiento, igualmente se hace uso del índice para agilizar la unión entre las tablas boleta y función, necesaria para este requerimiento. Mencionado anteriormente, este se implementa bajo un árbol B+. El usuario no hace parte de la llave primaria de la tabla Boletas, por lo tanto, no define la ubicación de las tuplas de esta haciéndolo un índice secundario.

### **Diseño de Datos**

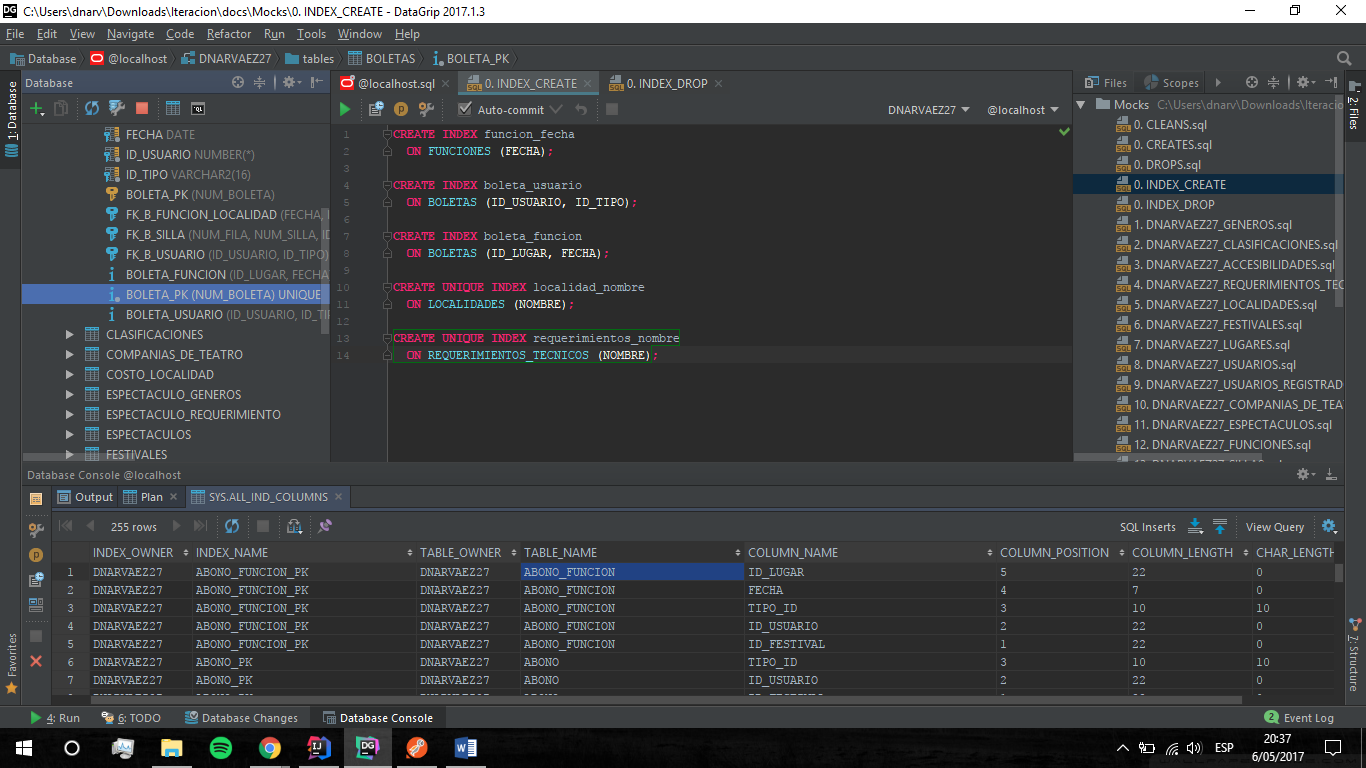


Ilustración 1

En la *ilustración 1* se muestra la información relevante a los índices en la Base de Datos, tanto la creación de estos, como información de Metadatos sobre los índices. Además, se observa que Oracle crea un índice único sobre las llaves primarias haciendo que estas sean de fácil acceso al ser requeridas.

El principalmente, las llaves primarias juegan un papel importante en la recuperación de información, dado que estas son la forma de obtener la información dada una llave. Es por esto que Oracle crea por defecto un índice para que la obtención de estos datos sea más eficiente. Además, varios *contraints* son también causa de que Oracle cree índices como *Unique*.

### **Documentación**

#### **RFC9**

* + **Escenarios de Prueba**
    - **Sentencia SQL**

SELECT DISTINCT

U.\*,

UR.\*

FROM FUNCIONES F INNER JOIN ESPECTACULOS E

ON F.ID\_ESPECTACULO = E.ID

INNER JOIN OFRECE O

ON O.ID\_ESPECTACULO = E.ID

INNER JOIN COMPANIAS\_DE\_TEATRO CT

ON O.ID\_COMPANIA\_DE\_TEATRO = CT.ID AND O.TIPO\_ID = CT.TIPO\_ID

INNER JOIN BOLETAS B

ON B.ID\_LUGAR = F.ID\_LUGAR AND B.FECHA = F.FECHA

INNER JOIN USUARIOS U

ON B.ID\_USUARIO = U.IDENTIFICACION AND B.ID\_TIPO = U.TIPO\_IDENTIFICACION

INNER JOIN USUARIOS\_REGISTRADOS UR

ON U.IDENTIFICACION = UR.ID\_USUARIO

AND U.TIPO\_IDENTIFICACION = UR.TIPO\_ID

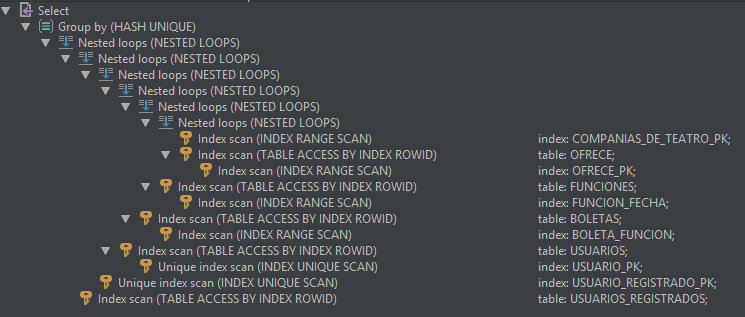
WHERE CT.ID = &ID

AND F.FECHA BETWEEN &F\_INICIO AND& F\_FIN;

* + - **Distribución de Datos**
    - **Valores de Parámetros**

|  |  |
| --- | --- |
| **Nombre Parámetro** | **Valor** |
| ID | 0 |
| F\_INICIO | TO\_DATE( '2014-10-14 00:56', 'YYYY-MM-DD HH24:MI' ) |
| F\_FIN | TO\_DATE( '2017-12-14 10:43', 'YYYY-MM-DD HH24:MI' ) |

* + - **Plan de Consulta**



* + - **Tiempo de Ejecución**
  + **Análisis de Eficiencia**

#### **RFC10**

* + **Escenarios de Prueba**
    - **Sentencia SQL**

SELECT \*

FROM USUARIOS

INNER JOIN USUARIOS\_REGISTRADOS UR

ON USUARIOS.IDENTIFICACION = UR.ID\_USUARIO AND USUARIOS.TIPO\_IDENTIFICACION = UR.TIPO\_ID

MINUS ( SELECT

U.\*,

UR.\*

FROM

FUNCIONES F INNER JOIN

ESPECTACULOS E

ON

F.ID\_ESPECTACULO = E.ID

INNER JOIN

OFRECE O

ON O.ID\_ESPECTACULO = E.ID

INNER JOIN

COMPANIAS\_DE\_TEATRO CT

ON O.ID\_COMPANIA\_DE\_TEATRO = CT.ID

AND O.TIPO\_ID = CT.TIPO\_ID

INNER JOIN

BOLETAS B

ON B.ID\_LUGAR = F.ID\_LUGAR AND B.FECHA = F.FECHA

INNER JOIN

USUARIOS U

ON B.ID\_USUARIO = U.IDENTIFICACION AND B.ID\_TIPO = U.TIPO\_IDENTIFICACION

INNER JOIN

USUARIOS\_REGISTRADOS UR

ON

U.IDENTIFICACION = UR.ID\_USUARIO AND U.TIPO\_IDENTIFICACION = UR.TIPO\_ID

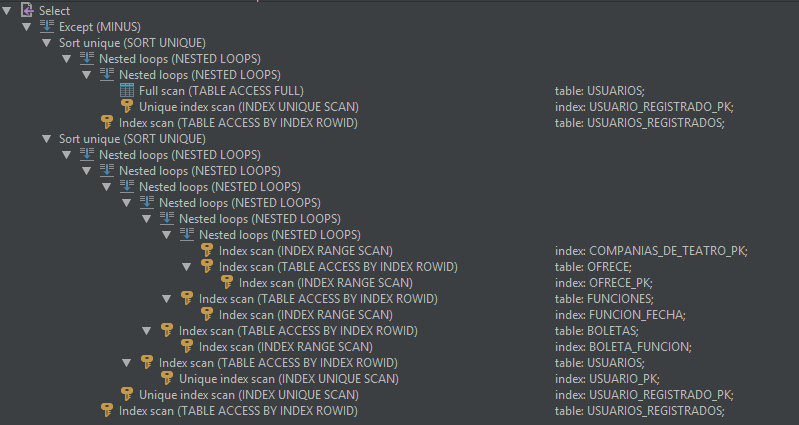
WHERE CT.ID = &ID

AND F.FECHA BETWEEN &F\_INICIO AND &F\_FIN );

* + - **Distribución de Datos**
    - **Valores de Parámetros**

|  |  |
| --- | --- |
| **NOMBRE PARAMETRO** | **VALOR** |
| ID | 0 |
| F\_INICIO | TO\_DATE( '2008-06-12 05:23', 'YYYY-MM-DD HH24:MI' ) |
| F\_FIN | TO\_DATE( '2030-08-18 01:50', 'YYYY-MM-DD HH24:MI' ) |

* + - **Plan de Consulta**



* + - **Tiempo de Ejecución**
  + **Análisis de Eficiencia**

#### **RFC11**

* + **Escenarios de Prueba**
    - **Sentencia SQL**

SELECT DISTINCT

E.NOMBRE AS ESPECTACULO,

Z.FECHA,

L.NOMBRE AS LUGAR,

Z.ROL,

NUM\_BOLETAS

FROM ( SELECT

F.\*,

U.ROL,

COUNT( \* ) AS NUM\_BOLETAS

FROM FUNCIONES F INNER JOIN BOLETAS B

ON B.FECHA = F.FECHA AND B.ID\_LUGAR = F.ID\_LUGAR

INNER JOIN USUARIOS U

ON U.IDENTIFICACION = B.ID\_USUARIO AND U.TIPO\_IDENTIFICACION = B.ID\_TIPO

INNER JOIN LOCALIDADES L

ON B.ID\_LOCALIDAD = L.ID

WHERE L.NOMBRE = &NOMBRE\_LOCALIDAD

GROUP BY F.FECHA, F.ID\_LUGAR, ID\_ESPECTACULO, SE\_REALIZA, ROL ) Z INNER JOIN ESPECTACULOS E

ON E.ID = Z.ID\_ESPECTACULO

INNER JOIN ESPECTACULO\_REQUERIMIENTO ER

ON E.ID = ER.ID\_ESPECTACULO

INNER JOIN REQUERIMIENTOS\_TECNICOS RT

ON ER.ID\_REQUERIMIENTO = RT.ID

INNER JOIN LUGARES L

ON L.ID = Z.ID\_LUGAR

WHERE 1 = 1 AND ( RT.NOMBRE = &REQ\_1 OR RT.NOMBRE = &REQ\_2 )

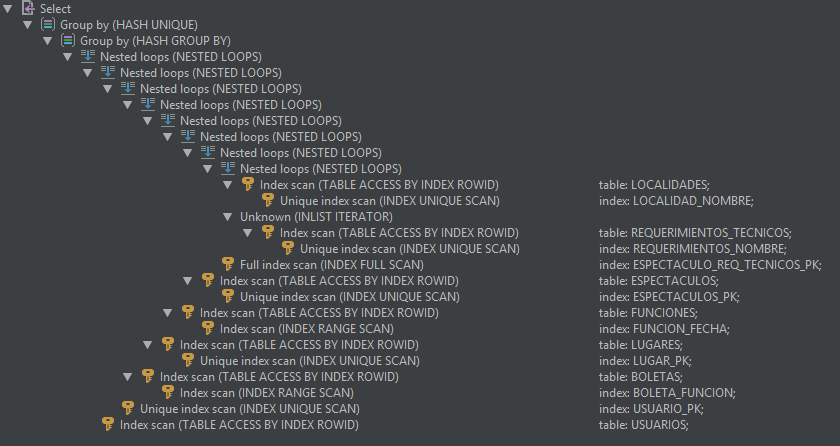
AND TO\_CHAR( Z.FECHA, 'HH24:MI' ) BETWEEN &H\_INI AND &H\_FIN

AND Z.FECHA BETWEEN &F\_INI AND &F\_FIN;

* + - **Distribución de Datos**
    - **Valores de Parámetros**

|  |  |
| --- | --- |
| **NOMBRE PARAMETRO** | **VALOR** |
| NOMBRE\_LOCALIDAD | VIP |
| REQ\_1 | Agua |
| REQ\_2 | Efecto de Luces |
| H\_INI | 14:00 |
| H\_FIN | 20:00 |
| F\_INI | TO\_DATE( '2017-07-12 17:53', 'YYYY-MM-DD HH24:MI' ) |
| F\_FIN | TO\_DATE( '2020-09-12 03:40', 'YYYY-MM-DD HH24:MI' ) |

* + - **Plan de Consulta**



* + - **Tiempo de Ejecución**
  + **Análisis de Eficiencia**

#### **RFC12**

* + **Escenarios de Prueba**
    - **Sentencia SQL**

SELECT

U.\*,

UR.\*,

NUM\_BOLETAS\_VIP,

NUM\_BOLETAS

FROM ( SELECT

ID\_USUARIO,

ID\_TIPO,

ID\_LOCALIDAD,

COUNT( \* ) AS NUM\_BOLETAS\_VIP

FROM BOLETAS B

WHERE ID\_LOCALIDAD = &ID\_LOCALIDAD

GROUP BY ID\_USUARIO, ID\_TIPO, ID\_LOCALIDAD ) Z INNER JOIN LOCALIDADES L

ON ID\_LOCALIDAD = L.ID

INNER JOIN ( SELECT

ID\_USUARIO,

ID\_TIPO,

SUM( NUM\_BOLETAS ) AS NUM\_BOLETAS

FROM ( SELECT

ID\_USUARIO,

ID\_TIPO,

ID\_LOCALIDAD,

COUNT( \* ) AS NUM\_BOLETAS

FROM BOLETAS B

GROUP BY ID\_USUARIO, ID\_TIPO, ID\_LOCALIDAD ) Z INNER JOIN LOCALIDADES L

ON ID\_LOCALIDAD = L.ID

GROUP BY ID\_USUARIO, ID\_TIPO ) W

ON Z.ID\_USUARIO = W.ID\_USUARIO AND Z.ID\_TIPO = W.ID\_TIPO

INNER JOIN USUARIOS U

ON U.IDENTIFICACION = Z.ID\_USUARIO AND U.TIPO\_IDENTIFICACION = Z.ID\_TIPO

INNER JOIN USUARIOS\_REGISTRADOS UR

ON U.IDENTIFICACION = UR.ID\_USUARIO AND U.TIPO\_IDENTIFICACION = UR.TIPO\_ID

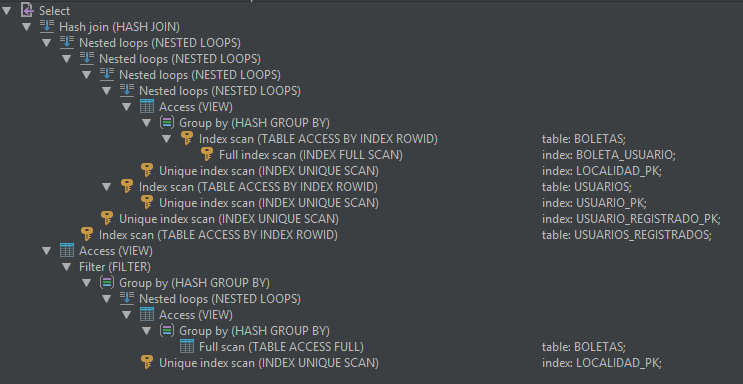
WHERE NUM\_BOLETAS\_VIP = NUM\_BOLETAS

AND NUM\_BOLETAS >= &NUMERO\_BOLETAS;

* + - **Distribución de Datos**
    - **Valores de Parámetros**

|  |  |
| --- | --- |
| **NOMBRE PARAMETRO** | **VALOR** |
| ID\_LOCALIDAD | 1 |
| NUMERO\_BOLETAS | 0 |

* + - **Plan de Consulta**



* + - **Tiempo de Ejecución**
  + **Análisis de Eficiencia**