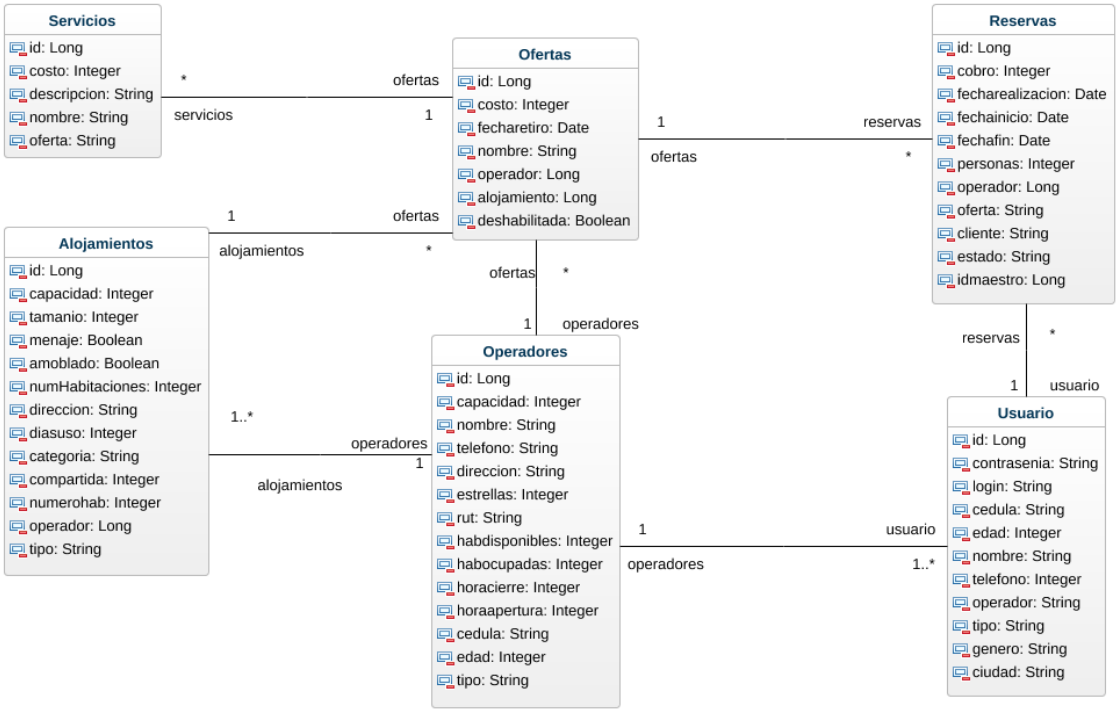
Alohandes – Iteración 3 – Diseño físico y optimización de consultas

Diana Vanessa Gonzalez – Ivan D. Salazar
Universidad de los Andes, Bogotá, Colombia
dv.gonzalez10, id.salazar@uniandes.edu.co
fecha de presentación: 24/04/2018

Tabla de Contenido

Alohandes – Iteración 3 – Diseño físico y optimización de consultas 1
1. Modelo conceptual 2
2. Diseño de la aplicación 2
Diseño físico: 2
3. Construcción de la aplicación 6
Diseño del escenario de pruebas de eficiencia: 6
Ajuste de las tablas creadas en Oracle 6
Ajuste de las clases involucradas en los nuevos requerimientos 7
Análisis del proceso de optimización y el modelo de ejecución de consultas 7

1. Modelo conceptual



Para nuestro nuevo modelo, nos basamos en el modelo de la iteración anterior. No creamos ninguna tabla nueva. La tabla de usuarios recibió modificaciones que obedecían a los nuevos requerimientos de la fase 3, por lo que agregamos género y ciudad, valores que antes no consideramos, para poder crear filtros pertinentes para los requerimientos 12 y 13. El resto de tablas no tuvieron cambio alguno

2. Diseño de la aplicación

Diseño físico:

- Para la tabla de reservas elegimos el índice no-clúster debido a que para este la información repetida no es aceptada. Por lo tanto los índices son secundarios ya que no tienen la primary key y puede tener información duplicada, como:
  Create index reservas_ofertas on reservas (oferta);
  Esta se organiza por ofertas, pero hay más de una oferta por reserva, por lo cual esta repetida.

- Los índices encontrados para nuestro modelo de datos, son creados automáticamente al crear una restricción PRIMARY KEY por Oracle. Estos índices son de tipo único, por lo que de la misma manera hacen cumplir la restricción en la que no hay valores

repetidos dentro de la misma columna, también proporciona un alto grado de unicidad. Este índice es clúster, así mismo ayuda a que el proceso de búsqueda se aceleré, accediendo a los registros de una manera más rápida. Un índice único proporciona información adicional al optimizador de consultas, además de que se encuentra en orden ascendente por lo que se sabe fácilmente recorrer la tabal para realizar una búsqueda, de lo contrario al agregar la información a la tabla se guardaría en orden aleatorio, y al realizar una búsqueda tocaría recorrerla toda, y no para solamente al encontrar una coincidencia con la búsqueda, si no que la sigue recorriendo, en caso de encontrar más.

Página Inicial | ivan | PK_OFERTAS

Columnas | Detalles | Estadísticas | Particiones | SQL

Acciones...

	INDEX_OWNER	INDEX_NAME	TABLE_OWNER	TABLE_NAME	COLUMN_NAME	COLUMN_POSITION	DESCEND
1	ISIS2304A651810	PK_OFERTAS	ISIS2304A651810	OFERTAS	ID	1	ASC

Página Inicial | ivan | PK_OPERADOR

Columnas | Detalles | Estadísticas | Particiones | SQL

Acciones...

	INDEX_OWNER	INDEX_NAME	TABLE_OWNER	TABLE_NAME	COLUMN_NAME	COLUMN_POSITION	DESCEND
1	ISIS2304A651810	PK_OPERADOR	ISIS2304A651810	OPERADORES	ID	1	ASC

Página Inicial | ivan | PK_RESERVAS

Columnas | Detalles | Estadísticas | Particiones | SQL

Acciones...

	INDEX_OWNER	INDEX_NAME	TABLE_OWNER	TABLE_NAME	COLUMN_NAME	COLUMN_POSITION	DESCEND
1	ISIS2304A651810	PK_RESERVAS	ISIS2304A651810	RESERVAS	ID	1	ASC

Página Inicial | ivan | PK_SERVICIOS

Columnas | Detalles | Estadísticas | Particiones | SQL

Acciones...

	INDEX_OWNER	INDEX_NAME	TABLE_OWNER	TABLE_NAME	COLUMN_NAME	COLUMN_POSITION	DESCEND
1	ISIS2304A651810	PK_SERVICIOS	ISIS2304A651810	SERVICIOS	ID	1	ASC

Página Inicial | ivan | PK_USUARIO

Columnas | Detalles | Estadísticas | Particiones | SQL

Acciones...

	INDEX_OWNER	INDEX_NAME	TABLE_OWNER	TABLE_NAME	COLUMN_NAME	COLUMN_POSITION	DESCEND
1	ISIS2304A651810	PK_USUARIO	ISIS2304A651810	USUARIOS	ID	1	ASC

Página Inicial | ivan | SYS_C00375311

Columnas | Detalles | Estadísticas | Particiones | SQL

Acciones...

	INDEX_OWNER	INDEX_NAME	TABLE_OWNER	TABLE_NAME	COLUMN_NAME	COLUMN_POSITION	DESCEND
1	ISIS2304A651810	SYS_C00375311	ISIS2304A651810	ALOJAMIENTOS	ID	1	ASC

Documentación para cada requerimiento

- RFC10-CONSULTAR CONSUMO EN ALOHANDES

  Requerimiento sin ningún índice aparte de los generados por Oracle, tiene una duración de 0,444segundos fue consultado con la oferta numero : 10000, también se definió la fecha entre el primero de enero del 2010 y el 2020 en este caso tiene 5 clientes que han realizado al menos una reserva durante el tiempo determinado para la oferta asignada

Hoja de Trabajo | Generador de Consultas

```
Select distinct usuarios.* from
(Select * from
(Select * from reservas where oferta = 10000)
filtro where FECHAFIN > '1/1/2010' and FECHAINICIO < '1/1/2020')
utiles join usuarios
on utiles.cliente = usuarios.id
order by genero;
```

Salida de Script | Resultado de la Consulta

SQL | Todas las Filas Recuperadas | 5 en 0,444 segundos

	ID	CONTRASENIA	LOGIN	CEDULA	EDAD	NOMBRE	TELEFONO	GENERO	CIUDAD	OPERADOR	TIPO
1	20000	contrasenia20000	login20000	20000	23	persona20000	3100019999	F	Bogota	(null)	cliente
2	15000	contrasenia15000	login15000	15000	23	persona15000	3100014999	F	Florencia	(null)	cliente
3	10000	contrasenia10000	login10000	10000	23	persona10000	3100009999	F	Duitama	(null)	cliente
4	50000	contrasenia50000	login50000	50000	23	persona50000	3100049999	F	Bogota	(null)	cliente
5	65000	contrasenia65000	login65000	65000	23	persona65000	3100064999	F	Bogota	(null)	cliente

  Pero al incorporar índice: Create index reservas_ofertas on reservas (oferta); que organiza a las reservas por ofertas se demora: 0,136

Hoja de Trabajo | Generador de Consultas

```
Select distinct usuarios.* from
(Select * from
(Select * from reservas where oferta = 10000)
filtro where FECHAFIN > '1/1/2010' and FECHAINICIO < '1/1/2020')
utiles join usuarios
on utiles.cliente = usuarios.id
order by genero;
```

Resultado de la Consulta

SQL | Todas las Filas Recuperadas | 5 en 0,136 segundos

	ID	CONTRASENIA	LOGIN	CEDULA	EDAD	NOMBRE	TELEFONO	GENERO	CIUDAD	OPERADOR	TIPO
1	20000	contrasenia20000	login20000	20000	23	persona20000	3100019999	F	Bogota	(null)	cliente
2	15000	contrasenia15000	login15000	15000	23	persona15000	3100014999	F	Florencia	(null)	cliente
3	10000	contrasenia10000	login10000	10000	23	persona10000	3100009999	F	Duitama	(null)	cliente
4	50000	contrasenia50000	login50000	50000	23	persona50000	3100049999	F	Bogota	(null)	cliente
5	65000	contrasenia65000	login65000	65000	23	persona65000	3100064999	F	Bogota	(null)	cliente

- RFC10 - CONSULTAR CONSUMO EN ALOHANDES – RFC10-V2

  Requerimiento sin ningún índice aparte de los generados por Oracle, tiene una duración de 0,444segundos fue consultado con la oferta numero : 10000, en este caso tiene 5 clientes que han realizado al menos una reserva

Hoja de Trabajo | Generador de Consultas

```
Select distinct usuarios.* from
(Select * from
(Select * from reservas where oferta = 2)
filtro where FECHAFIN > '1/1/2018' and FECHAINICIO < '1/1/2020')
utiles right outer join usuarios
on utiles.cliente = usuarios.id
where utiles.id is null and tipo = 'cliente'
order by genero;
```

Salida de Script | Resultado de la Consulta

SQL | Se han recuperado 50 filas en 0,697 segundos

	ID	CONTRASENIA	LOGIN	CEDULA	EDAD	NOMBRE	TELEFONO	GENERO	CIUDAD	OPERADOR	TIPO
1	99970	contrasenia99970	login99970	99970	53	persona99970	3100099969	F	Duitama	(null)	cliente
2	99972	contrasenia99972	login99972	99972	42	persona99972	3100099971	F	Florencia	(null)	cliente
3	99974	contrasenia99974	login99974	99974	48	persona99974	3100099973	F	Bogota	(null)	cliente
4	99976	contrasenia99976	login99976	99976	50	persona99976	3100099975	F	Duitama	(null)	cliente
5	99978	contrasenia99978	login99978	99978	19	persona99978	3100099977	F	Florencia	(null)	cliente
6	9998	contrasenia9998	login9998	9998	38	persona9998	3100009997	F	Bogota	(null)	cliente
7	99980	contrasenia99980	login99980	99980	20	persona99980	3100099979	F	Bogota	(null)	cliente
8	99982	contrasenia99982	login99982	99982	32	persona99982	3100099981	F	Duitama	(null)	cliente
9	99984	contrasenia99984	login99984	99984	20	persona99984	3100099983	F	Florencia	(null)	cliente

Con el índex, el tiempo se reduce a la mitad

Hoja de Trabajo | Generador de Consultas

```
Select distinct usuarios.* from
(Select * from
(Select * from reservas where oferta = 2)
filtro where FECHAFIN > '1/1/2018' and FECHAINICIO < '1/1/2020')
utiles right outer join usuarios
on utiles.cliente = usuarios.id
where utiles.id is null and tipo = 'cliente'
order by genero;
```

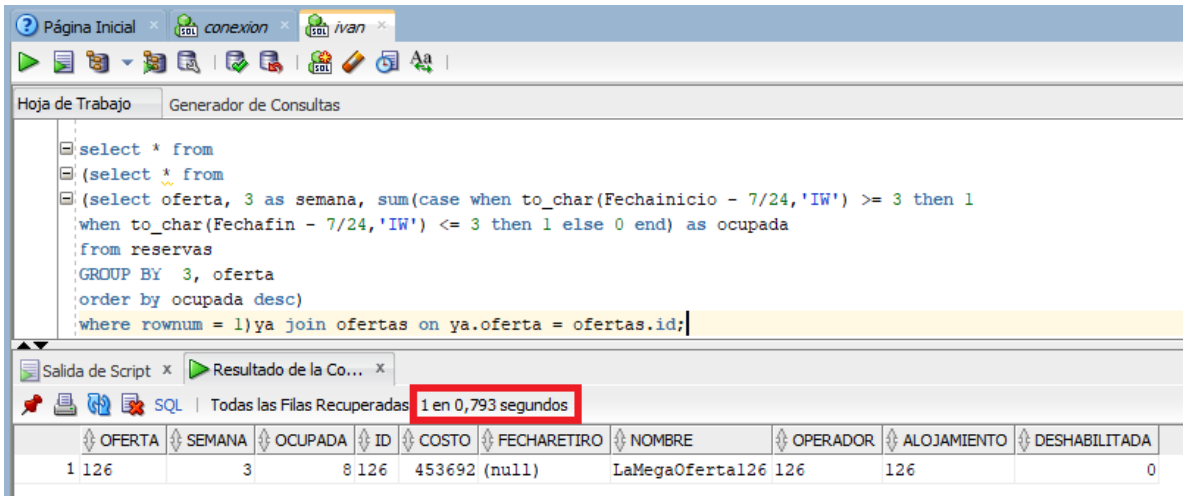
Salida de Script | Resultado de la Consulta

SQL | Se han recuperado 50 filas en 0,377 segundos

	ID	CONTRASENIA	LOGIN	CEDULA	EDAD	NOMBRE	TELEFONO	GENERO	CIUDAD	OPERADOR	TIPO
1	99970	contrasenia99970	login99970	99970	53	persona99970	3100099969	F	Duitama	(null)	cliente
2	99972	contrasenia99972	login99972	99972	42	persona99972	3100099971	F	Florencia	(null)	cliente
3	99974	contrasenia99974	login99974	99974	48	persona99974	3100099973	F	Bogota	(null)	cliente
4	99976	contrasenia99976	login99976	99976	50	persona99976	3100099975	F	Duitama	(null)	cliente
5	99978	contrasenia99978	login99978	99978	19	persona99978	3100099977	F	Florencia	(null)	cliente
6	9998	contrasenia9998	login9998	9998	38	persona9998	3100009997	F	Bogota	(null)	cliente
7	99980	contrasenia99980	login99980	99980	20	persona99980	3100099979	F	Bogota	(null)	cliente
8	99982	contrasenia99982	login99982	99982	32	persona99982	3100099981	F	Duitama	(null)	cliente
9	99984	contrasenia99984	login99984	99984	20	persona99984	3100099983	F	Florencia	(null)	cliente
10	99986	contrasenia99986	login99986	99986	23	persona99986	3100099985	F	Bogota	(null)	cliente

- RFC11 - CONSULTAR FUNCIONAMIENTO

  Se demora un total de 0,79 segundos, se da la información por semana según se necesite



- RFC12 - CONSULTAR LOS BUENOS CLIENTES

3. Construcción de la aplicación

Diseño del escenario de pruebas de eficiencia:

Para poblar las tablas de datos generamos las tablas en Excel. Para esto combinamos simple secuencia para datos que no eran tan relevantes y para los ids, y algunos datos aleatorios generados en Mockaroo para valores que queríamos controlar de alguna manera o cuyo tamaño no era tan grande. Luego de eso simplemente realizamos la inserción manual de los datos por medio de SQL Developer, ya que el archivo .key nunca funcionó. Para eso utilizamos la opción de importar data desde una fuente externa.
La población actual de las tablas equivale a 5000 Operadores, 105000 Usuarios, 100000 Alojamientos, 100000 Ofertas, 250000 Servicios y 400000 Reservas. Para un total de 960000 tuplas distribuidas en todas las tablas que componen la base de datos del problema.
Los documentos en Excel que pueblan las tablas se encuentran dentro de la carpeta Población, en los documentos del proyecto.

Ajuste de las tablas creadas en Oracle

Los requerimientos funcionales se encuentran dentro del proyecto

Ajuste de las clases involucradas en los nuevos requerimientos

Se realizaron los cambios correspondientes a los servicios REST para cumplir con los nuevos requerimientos, también se hicieron los cambios en la clase transaction manager y en los DAO correspondientes. Todo se encuentra dentro de la aplicación.

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

El sistema manejador de consultas maneja un conjunto de datos para convertirlos en información relevante, en el caso de Oracle esta preconfigurada lista para ser usada en producción, administración de almacenamiento y de memoria, entre otras aplicaciones. Oracle posee la capacidad de optimizar las consultas que requieren el manejo de Joins, selecciones y otros requerimientos similares. La base de datos de Oracle, utiliza los índices y los algoritmos de unión para reducir la cantidad de memoria que se utiliza. Esta reducción se da porque al tener almacenado cierto orden de la información Oracle puede decidir con mayor facilidad en qué lugar se encuentra la información que se necesita buscar. Esto evita gastar tiempo de procesamiento y memoria disponible al traer y analizar datos innecesarios o que de entrada se sabe que son impertinentes. Cosa que haría un algoritmo que simple y llanamente analiza absolutamente toda la información.