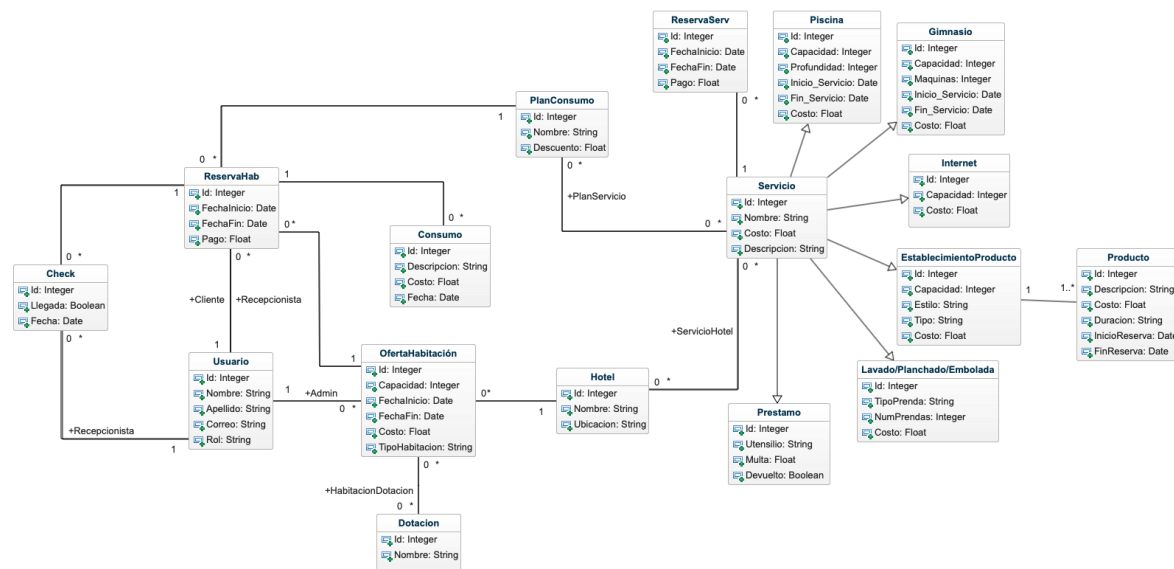


Iteración 02: Documento de Análisis

Introducción

En la actualidad la industria hotelera es uno de los sectores de la economía más importantes para el PIB y la economía mundial. Los hoteles y diferentes alternativas de alojamiento ofrecen diferentes servicios y comodidades para los clientes. Tenemos como objetivo tratar de crear un SGBD para poder manejarnos en este sector, buscando cumplir los 11 requerimientos funcionales junto a los 3 requerimientos no funcionales. Todo lo anterior mediante el uso de las siguientes tecnologías: Java, Oracle, Spring Boot y Maven, con las cuales podremos evidenciar el cumplimiento de los mismos y podremos a prueba nuestros modelos (E/R, UML y relacional) con el fin de ver que el planteamiento sea correcto y sea posible ejecutarlos en un entorno funcional.

Modelo UML Corregido:



Modelo Relacional Corregido:

OfertasHabitaciones							
ID	CAPACIDAD	FECHAINICIO	FECHAFIN	COSTO	TIPOHABITACION	IDUSUARIO	IDHOTEL
ND,NN,SA,PK	NN,CK[0..10]	NN,SA	NN,SA	NN,SA	NN,SA	NN,SA,FK	NN,SA,FK
1	6	10/11/2022	20/11/2022	\$2.000.000,00	Suite Familiar	1543	1
2	2	20/12/2022	23/12/2022	\$500.000,00	Doble	1543	2

Universidad de Los Andes

Sistemas Transaccionales

Iteración 02: Documento de Análisis

Juliana Sofía Ahumada Arcos – 201921471

Juan Felipe Caraballo Umbarila – 201923741

Sebastián Umaña Peinado – 202013778

Dotaciones

ID	NOMBRE
NN,ND,PK	NN,ND
1	televisor
2	cocina
3	nevera
4	mini bar

HabitacionesDotaciones

IDHABITACION	IDDOTACION
NN,ND,FK	NN,ND,FK
1	1
2	2
1	3
2	4

ReservasHabitaciones

ID	FECHAINICIO	FECHAFIN	PAGO	IDPLANCONSUMO	IDOFERTA	HABITACION	IDUSUARIO
NN,ND,SA,PK	NN,UA	NN,UA	NN,DD	NN,FK	NN,FK	NN,FK	NN,FK
30	11/11/2022	13/11/2022	\$6.000.000,00	10	1		1540
31	20/12/2022	21/12/2022	\$1.000.000,00	11	2		1540

Hoteles

ID	NOMBRE	UBICACION
NN,ND,PK,SA	NN, UA	NN, UA
1	DeCameron	12.000
2	Las Islas	34.000

ServicioHotel

IDHOTEL	IDSERV
PK, FK(hoteles.id)	PK, FK(servicios.id)
1	1
2	2

PlanesConsumo

ID	NOMBRE	DESCUENTO
NN,ND,PK,SA	NN, UA	NN, UA
1	Larga Estadia	20.000
2	Tiempo Compartido	15.000

PlanServicio

IDPLAN	IDSERV
FK(planesconsumo.id)	FK(servicios.id)
1	1
2	2

Universidad de Los Andes

Sistemas Transaccionales

Iteración 02: Documento de Análisis

Juliana Sofía Ahumada Arcos – 201921471

Juan Felipe Caraballo Umbarila – 201923741

Sebastián Umaña Peinado – 202013778

Usuarios

ID	NOMBRE	APELLIDO	CORREO	ROL
ND,NN,SA,PK	NN,UA	NN,UA	NN,UA	NN,SA,CK["Cliente","Recepcionista","Empleado","Admin","Gerente"]
1540	Carlos	Gonzales	cg@gmail.com	Cliente
1541	Juan	Garcia	lg@hotmail.com	Recepcionista
1542	Maria	Lopez	ml@yahoo.com	Empleado
1543	Esteban	Rodriguez	er@hotmail.com	Admin
1544	Pedro	Gomez	pg@hotmail.com	Gerente

Checks

ID	LLEGADA	FECHA	IDUSUARIO	IDRESERVA
NN,ND,PK, SA	NN	NN,UA	NN,SA,FK(usuarios.id)	NN,SA,FK(reservas.id)
10	VERDADERO	11/12/2023	1541	20
11	FALSO	12/10/2023	1541	21

Servicios

ID	NOMBRE	COSTO	DESCRIPCION
NN,ND,PK,SA	NN,UA	NN,UA	NN,UA
1	SPA	12.000	Es un servicio que ofrece...
2	BAR	34.000	Es un servicio que ofrece...

Ofrecen

ESTABLECIMIENTO	PRODUCTOS_ID
entos.id)	d)
10	1

Productos

ID	COSTO	DESCRIPCION
NN,ND,PK	NN, UA	NN, UA
10	1000	Almuerzo

Consumos

ID	DESCRIPCION	COSTO	FECHA
NN,ND,PK	NN, UA	NN, UA	NN, UA
1	coca-cola	12.000	25/09/2021
2	almuerzo	45.000	24/07/2024

ReservasServ

ID	FECHAINICIO	FECHAFIN	PAGO
NN,ND,PK	NN, UA	NN, UA	NN, UA
1	24/05/2022	25/05/2022	50.000
2	23/04/2021	24/04/2021	45.000

Universidad de Los Andes

Sistemas Transaccionales

Iteración 02: Documento de Análisis

Juliana Sofía Ahumada Arcos – 201921471

Juan Felipe Caraballo Umbarila – 201923741

Sebastián Umaña Peinado – 202013778

Piscinas

ID	CAPACIDAD	PROFUNDIDAD	INICIO_SERVICIO	FIN_SERVICIO	COSTO
NN,ND,SA,PK	NN,UA	NN,UA	NN,UA	NN,UA	NN,UA
1	30	2,5	8:00:00 a. m.	8:00:00 p. m.	\$50.000
2	35	1,5	9:00:00 a. m.	9:00:00 p. m.	\$45.000

Gimnasios

ID	CAPACIDAD	MAQUINAS	INICIO_SERVICIO	FIN_SERVICIO	COSTO
NN,ND,SA,PK	NN,UA	NN,UA	NN,UA	NN,UA	NN,UA
1	30	2,5	8:00:00 a. m.	8:00:00 p. m.	\$50.000
2	35	1,5	9:00:00 a. m.	9:00:00 p. m.	\$45.000

Internet

ID	CAPACIDAD	COSTO
NN,ND,SA,PK	NN,UA	NN,UA
1	30	\$50.000
2	35	\$45.000

EstablecimientosProductos

ID	CAPACIDAD	ESTILO	Costo	TIPO
NN,ND,SA,PK	UA	UA	UA	NN,UA, CK(Bar, Restaurante, Salon, Supermercado, Tienda, SPA)
1	30	null	\$50.000	Salon
2	35	Rock	null	Restaurante

Salones

ID	CAPACIDAD	COSTO
NN,ND,SA,PK	NN,UA	NN,UA
1	30	\$50.000
2	35	\$45.000

Prestamos

ID	UTENSILIO	MULTA	DEVUELTO
NN,ND,SA,PK	NN,UA	NN,UA	NN,UA, CK(SI, NO)
1	Toalla	\$50.000	SI
2	Toalla	\$45.000	NO

Lavados/Planchados/Emboladas

ID	TIPOPRENDA	NUMPRENDAS	COSTO
NN,ND,SA,PK	NN,UA	NN,UA	NN,UA
1	Camisa	2	\$50.000
2	Zapato	4	\$40.000

Universidad de Los Andes

Sistemas Transaccionales

Iteración 02: Documento de Análisis

Juliana Sofía Ahumada Arcos – 201921471

Juan Felipe Caraballo Umbarila – 201923741

Sebastián Umaña Peinado – 202013778

Productos

ID	DESCRIPCION	COSTO	DURACION	INICIORESERVA	FINRESERVA	SERVICIO
NN,ND,SA,PK	NN,UA	UA	UA	UA	UA	NN,UA
1	Masaje	\$150.000	8:00:00 a. m.	8:00:00 a. m.	8:45:00 a. m.	SPA
2	Aceite	\$40.000	null	null	null	Supermercado

Diseño de la Aplicación:

- RFC1: Mostrar el dinero recolectado por servicios en cada habitación en el último año corrido.

Sentencia SQL:

```
SELECT habitacion_id, sum(pago) as pago
FROM reservaservicios WHERE fechafin >= ADD_MONTHS(TRUNC(SYSDATE),-12) AND fechafin < TRUNC(SYSDATE)
GROUP BY habitacion_id
```

Teniendo en cuenta la sentencia SQL utilizada, no es necesario crear un índice, ya que a pesar de que la consulta incluye la función agregada sobre la columna pago, existe una sentencia where y una sentencia groupby sobre las columnas fechafin y habitación_id.

Plan de Ejecución:

OPERATION	OBJECT_NAME	OPTIONS	CARDINALITY	COST
SELECT STATEMENT				147
HASH				52812
GROUP BY				52812
TABLE ACCESS	RESERVASERVICIOS	FULL		52812
AND				145
Filter Predicates				
TRUNC(SYSDATE@0) > ADD_MONTHS(TRUNC(SYSDATE@0), (-12))				
Filter Predicates				
FECHAFIN < TRUNC(SYSDATE@0)				
FECHAFIN >= ADD_MONTHS(TRUNC(SYSDATE@0), (-12))				

- RFC2: Mostrar los 20 servicios más populares.

Sentencia SQL:

```
SELECT nombre_servicio, sum(existe) as existe
FROM reservaservicios WHERE fechafin <= :fecha2 AND fechafin >= :fecha1
GROUP BY nombre_servicio
ORDER BY existe DESC;
```

Teniendo en cuenta la sentencia SQL utilizada, no es necesario crear un índice, ya que a pesar de que la consulta incluye la función agregada sobre la columna existe, dentro de las sentencias SQL, existe una sentencia where y una sentencia groupby sobre las columnas fechafin y nombre_servicio.

Plan de Ejecución:

Universidad de Los Andes

Sistemas Transaccionales

Iteración 02: Documento de Análisis

Juliana Sofía Ahumada Arcos – 201921471

Juan Felipe Caraballo Umbarila – 201923741

Sebastián Umaña Peinado – 202013778

OPERATION	OBJECT_NAME	OPTIONS	CARDINALITY	COST
SELECT STATEMENT				146
SORT		ORDER BY	270	146
HASH		GROUP BY	270	146
FILTER			270	146
Filter Predicates				
TO_DATE(FECHA1) <= TO_DATE(FECHA2)				
TABLE ACCESS	RESERVASERVICIOS	FULL	270	144
Filter Predicates				

- RFC3: Mostrar el índice de ocupación de cada una de las habitaciones del hotel.

Sentencia SQL:

```
SELECT habitacion_id, sum(existe) as existe FROM ofertashabitaciones
WHERE fechafin >= ADD_MONTHS(TRUNC(SYSDATE),-12) AND fechafin < TRUNC(SYSDATE)
GROUP BY habitacion_id;
```

Teniendo en cuenta la sentencia SQL utilizada, no es necesario crear un índice, ya que a pesar de que la consulta incluye la función agregada sobre la columna existe, dentro de las sentencias SQL, existe una sentencia where y una sentencia groupby sobre las columnas fechafin y habitación_id.

Plan de Ejecución:

OPERATION	OBJECT_NAME	OPTIONS	CARDINALITY	COST
SELECT STATEMENT				171
HASH		GROUP BY	52334	171
FILTER				
Filter Predicates				
TRUNC(SYSDATE@) > ADD_MONTHS(TRUNC(SYSDATE@),(-12))				
TABLE ACCESS	OFERTASHABITACIONES	FULL	52334	168
Filter Predicates				
AND				
FECHAFIN < TRUNC(SYSDATE@)				
FECHAFIN >= ADD_MONTHS(TRUNC(SYSDATE@),(-12))				

- RFC4

Sentencia SQL:

```
--Filtro por fecha
SELECT s.id, s.nombre, s.costo, s.descripcion
FROM servicios s
JOIN reservaservicios rs ON s.id = rs.servicios_id
WHERE fechafin <= :fecha2 AND fechafin >= :fecha1;

--Filtro por precio
SELECT s.id, s.nombre, s.costo, s.descripcion
FROM servicios s
JOIN reservaservicios rs ON s.id = rs.servicios_id
WHERE lower(nombre) = lower(:nombre1);

--Filtro por categoría
SELECT s.id, s.nombre, s.costo, s.descripcion
FROM servicios s
JOIN reservaservicios rs ON s.id = rs.servicios_id
WHERE costo <= :costo2 AND costo >= :costo1;
```

Universidad de Los Andes

Sistemas Transaccionales

Iteración 02: Documento de Análisis

Juliana Sofía Ahumada Arcos – 201921471

Juan Felipe Caraballo Umbarila – 201923741

Sebastián Umaña Peinado – 202013778

Plan de Ejecución:

- Filtro por Fecha:

OPERATION	OBJECT_NAME	OPTIONS	CARDINALITY	COST
NESTED LOOPS	TO_DATE(:FECHA1)<=TO_DATE(:FECHA2)			
NESTED LOOPS				270 144
TABLE ACCESS	RESERVASERVICIOS	FULL		270 144
Filter Predicates				
AND				
RS.FECHAFIN<=:FECHA2				
RS.FECHAFIN>=:FECHA1				
INDEX	SERVICIOS_PK	UNIQUE SCAN	1	0
Access Predicates				
S.ID=RS.SERVICIOS_ID				
TABLE ACCESS	SERVICIOS	BY INDEX ROWID	1	0

- Filtro por Precio:

OPERATION	OBJECT_NAME	OPTIONS	CARDINALITY	COST
SELECT STATEMENT				717 148
FILTER				
Filter Predicates				
TO_NUMBER(:COSTO1)<=TO_NUMBER(:COSTO2)				
NESTED LOOPS				717 148
NESTED LOOPS				107874 148
TABLE ACCESS	RESERVASERVICIOS	FULL		107874 143
INDEX	SERVICIOS_PK	UNIQUE SCAN	1	0
Access Predicates				
S.ID=RS.SERVICIOS_ID				
TABLE ACCESS	SERVICIOS	BY INDEX ROWID	1	0
Filter Predicates				
AND				
S.COSTO<=TO_NUMBER(:COSTO2)				
S.COSTO>=TO_NUMBER(:COSTO1)				

- Filtro por Categoría:

OPERATION	OBJECT_NAME	OPTIONS	CARDINALITY	COST
SELECT STATEMENT				1086 148
NESTED LOOPS				1086 148
NESTED LOOPS				107874 148
TABLE ACCESS	RESERVASERVICIOS	FULL		107874 143
INDEX	SERVICIOS_PK	UNIQUE SCAN	1	0
Access Predicates				
S.ID=RS.SERVICIOS_ID				
TABLE ACCESS	SERVICIOS	BY INDEX ROWID	1	0
Filter Predicates				
LOWER(S.NOMBRE)=LOWER(:NOMBRE1)				

- RFC5: Mostrar el consumo en Hotel Andes por un usuario dado, en un rango de fechas indicado.

Sentencia SQL:

```
SELECT h.nombre AS hotel_nombre,  
       c.fecha AS fecha_consumo,  
       c.descripcion AS descripcion_consumo,  
       c.costo AS costo_consumo  
FROM hoteles h  
JOIN ofertashabitaciones oh ON h.id = oh.hoteles_id  
JOIN reservahabitaciones rh ON oh.id = rh.ofertashabitaciones_id  
JOIN consumos c ON rh.id = c.reservahabitaciones_id  
JOIN usuarios u ON rh.usuarios_id = u.id  
WHERE u.id = :id  
       AND c.fecha BETWEEN :fecha1 AND :fecha2;
```

Universidad de Los Andes

Sistemas Transaccionales

Iteración 02: Documento de Análisis

Juliana Sofía Ahumada Arcos – 201921471

Juan Felipe Caraballo Umbarila – 201923741

Sebastián Umaña Peinado – 202013778

Teniendo en cuenta la sentencia SQL utilizada, para optimizar la consulta se puede crear un índice secundario simple en la columna de fecha de la tabla consumos, ya que al momento de realizar el filtro `c.fecha BETWEEN :fecha1 AND :fecha2`, la consulta puede encontrar los registros que se encuentren dentro de la fecha indicada sin necesidad de realizar una exploración completa a la tabla. Los otros campos incluidos dentro de la sentencia ya tienen índices primarios simples creados por defecto.

Plan de Ejecución (Sin utilizar índices):

OPERATION	OBJECT_NAME	OPTIONS	CARDINALITY	COST
SELECT STATEMENT				245
FILTER				96
Filter Predicates				
TO_DATE(:FECHA2)>=TO_DATE(:FECHA1)				
NESTED LOOPS				245
NESTED LOOPS				96
NESTED LOOPS				245
NESTED LOOPS				245
TABLE ACCESS	CONSUMOS	FULL		245
Filter Predicates				96
AND				
C.FECHA>=:FECHA1				
C.FECHA<=:FECHA2				
TABLE ACCESS	RESERVAHABITACIONES	BY INDEX ROWID		1
Filter Predicates				0
RH.USUARIOS_ID=TO_NUMBER(:ID)				
INDEX	RESERVAHABITACIONES_PK	UNIQUE SCAN		1
Access Predicates				0
RH.ID=C.RESERVAHABITACIONES_ID				
TABLE ACCESS	OFERTASHABITACIONES	BY INDEX ROWID		1
INDEX	OFERTASHABITACIONES_PK	UNIQUE SCAN		1
Access Predicates				0
OH.ID=RH.OFERTASHABITACIONES_ID				
INDEX	HOTELES_PK	UNIQUE SCAN		1
Access Predicates				0
H.ID=OH.HOTELES_ID				
TABLE ACCESS	HOTELES	BY INDEX ROWID		1

Plan de Ejecución (Usando índices):

OPERATION	OBJECT_NAME	OPTIONS	CARDINALITY	COST
SELECT STATEMENT				245
FILTER				34
Filter Predicates				
TO_DATE(:FECHA2)>=TO_DATE(:FECHA1)				
NESTED LOOPS				245
NESTED LOOPS				34
NESTED LOOPS				245
NESTED LOOPS				245
TABLE ACCESS	CONSUMOS	BY INDEX ROWID BATCHED		245
INDEX	ID_CONSUMOS_FECHA	RANGE SCAN		441
Access Predicates				2
AND				
C.FECHA>=:FECHA1				
C.FECHA<=:FECHA2				
TABLE ACCESS	RESERVAHABITACIONES	BY INDEX ROWID		1
Filter Predicates				0
RH.USUARIOS_ID=TO_NUMBER(:ID)				
INDEX	RESERVAHABITACIONES_PK	UNIQUE SCAN		1
Access Predicates				0
RH.ID=C.RESERVAHABITACIONES_ID				
TABLE ACCESS	OFERTASHABITACIONES	BY INDEX ROWID		1
INDEX	OFERTASHABITACIONES_PK	UNIQUE SCAN		1
Access Predicates				0
OH.ID=RH.OFERTASHABITACIONES_ID				
INDEX	HOTELES_PK	UNIQUE SCAN		1
Access Predicates				0
H.ID=OH.HOTELES_ID				
TABLE ACCESS	HOTELES	BY INDEX ROWID		1

Universidad de Los Andes

Sistemas Transaccionales

Iteración 02: Documento de Análisis

Juliana Sofía Ahumada Arcos – 201921471

Juan Felipe Caraballo Umbarila – 201923741

Sebastián Umaña Peinado – 202013778

- RFC6: Analizar la operación de Hotel Andes.

Sentencia SQL:

```
--Mayor Ocupación
SELECT fechainicio, COUNT(*)
FROM reservahabitaciones
GROUP BY fechainicio
ORDER BY COUNT(*) DESC;

--Menor Ocupación
SELECT fechainicio, COUNT(*)
FROM reservahabitaciones
GROUP BY fechainicio
ORDER BY COUNT(*) ASC;

--Mayores Consumos
SELECT fecha, SUM(costo)
FROM consumos
GROUP BY fecha
ORDER BY SUM(costo) DESC;
```

Teniendo en cuenta las sentencias SQL utilizadas, se podría considerar generar un índice para la columna fechainicio de la tabla reservahabitaciones o para la columna fecha de la tabla consumos. Sin embargo, el costo no cambiaría, pues de igual forma se tendría que recorrer ambas tablas en su totalidad para contar el número de ocurrencias y ordenar la tabla teniendo en cuenta dicha cantidad. Algo similar ocurre para la última sentencia, teniendo en cuenta que se debe ordenar según el costo.

Plan de Ejecución:

- Mayor Ocupación:

OPERATION	OBJECT_NAME	OPTIONS	CARDINALITY	COST
SELECT STATEMENT			386924	428
SORT		ORDER BY	386924	428
HASH		GROUP BY	386924	428
TABLE ACCESS	RESERVAHABITACIONES	FULL	386924	405

- Menor Ocupación:

OPERATION	OBJECT_NAME	OPTIONS	CARDINALITY	COST
SELECT STATEMENT			386924	428
SORT		ORDER BY	386924	428
HASH		GROUP BY	386924	428
TABLE ACCESS	RESERVAHABITACIONES	FULL	386924	405

Universidad de Los Andes

Sistemas Transaccionales

Iteración 02: Documento de Análisis

Juliana Sofía Ahumada Arcos – 201921471

Juan Felipe Caraballo Umbarila – 201923741

Sebastián Umaña Peinado – 202013778

- Mayores Consumos:

OPERATION	OBJECT_NAME	OPTIONS	CARDINALITY	COST
SELECT STATEMENT				97918 102
SORT		ORDER BY		97918 102
HASH		GROUP BY		97918 102
TABLE ACCESS	CONSUMOS	FULL		97918 95

- RFC7: Encontrar los buenos clientes.

Sentencia SQL:

```
SELECT u.id AS usuario_id, u.nombre AS nombre_usuario,
       u.apellido AS apellido_usuario,
       SUM(CASE WHEN r.fechafin >= SYSDATE - 365 THEN r.pago ELSE 0 END) AS total_pagos_ultimo_anio,
       SUM(CASE WHEN o.fechainicio >= SYSDATE - 365 THEN (o.fechafin - o.fechainicio) ELSE 0 END) AS total_dias_estadia_ultimo_anio
FROM usuarios u
LEFT JOIN reservahabitaciones r ON u.id = r.usuarios_id
LEFT JOIN ofertashabitaciones o ON r.ofertashabitaciones_id = o.id
GROUP BY u.id, u.nombre, u.apellido
HAVING SUM(CASE WHEN r.fechafin >= SYSDATE - 365 THEN r.pago ELSE 0 END) > 15000000
OR SUM(CASE WHEN o.fechainicio >= SYSDATE - 365 THEN (o.fechafin - o.fechainicio) ELSE 0 END) >= 14
```

Teniendo en cuenta la sentencia SQL utilizada, no es necesario crear un índice, ya que a pesar de que la consulta incluye la función agregada sobre la columna existe, dentro de las sentencias SQL, existe una sentencia where y una sentencia groupby sobre las columnas fechafin y habitación_id.

Plan de Ejecución:

OPERATION	OBJECT_NAME	OPTIONS	CARDINALITY	COST
SELECT STATEMENT				408614 2827
FILTER				
Filter Predicates				
OR				
SUM(CASE WHEN R.FECHAFIN>=SYSDATE@(-365 THEN R.PAGO ELSE 0 END)>15000000				
SUM(CASE WHEN O.FECHAINICIO>=SYSDATE@(-365 THEN O.FECHAFIN-O.FECHAINICIO ELSE 0 END)>=14				
HASH		GROUP BY	408614	2827
NESTED LOOPS		OUTER	408614	2815
HASH JOIN		RIGHT OUTER	408612	2797
Access Predicates				
U.ID=R.USUARIOS_ID(+)				
TABLE ACCESS	RESERVAHABITACIONES	FULL	386924	406
TABLE ACCESS	USUARIOS	FULL	103789	166
TABLE ACCESS	OFERTASHABITACIONES	BY INDEX ROWID	1	0
INDEX	OFERTASHABITACIONES_PK	UNIQUE SCAN	1	0
Access Predicates				
R.OFERTASHABITACIONES_ID=O.ID(+)				

- RFC8: Encontrar los servicios que no tienen mucha demanda.

Sentencia SQL:

```
SELECT s.id, s.nombre, COUNT(rs.id) AS veces_solicitado
FROM servicios s
LEFT JOIN reservaservicios rs ON s.id = rs.servicios_id
WHERE rs.fechainicio >= (SYSDATE - 365)
GROUP BY s.id, s.nombre
HAVING COUNT(rs.id) < 3;
```

Universidad de Los Andes

Sistemas Transaccionales

Iteración 02: Documento de Análisis

Juliana Sofía Ahumada Arcos – 201921471

Juan Felipe Caraballo Umbarila – 201923741

Sebastián Umaña Peinado – 202013778

Teniendo en cuenta la sentencia SQL utilizada, no es necesario crear un índice, ya que la sentencia debe recorrer toda la tabla para contar las veces que aparece el id de reservaservicios. Adicionalmente, existen las sentencias where y groupby realizadas sobre columnas distintas.

Plan de Ejecución:

OPERATION	OBJECT_NAME	OPTIONS	CARDINALITY	COST
SELECT STATEMENT			107874	154
FILTER				
Filter Predicates				
COUNT(*)<3				
HASH		GROUP BY	107874	154
NESTED LOOPS			107874	150
NESTED LOOPS			107874	150
TABLE ACCESS	RESERVASERVICIOS	FULL	107874	146
Filter Predicates				
RS.FECHAINICIO>=SYSDATE@!-365				
INDEX	SERVICIOS_PK	UNIQUE SCAN	1	0
TABLE ACCESS	SERVICIOS	BY INDEX ROWID	1	0

- RFC9

Sentencia SQL:

```
SELECT rs.id, rs.fechafin, rs.nombre_servicio, u.id, u.nombre FROM
RESERVASERVICIOS rs JOIN
OFERTASHABITACIONES oh ON rs.habitacion_id = oh.id
JOIN USUARIOS u ON u.id = oh.usuarios_id
WHERE rs.fechainicio >= :fecha1 AND rs.fechafin <= :fecha2;
```

Plan de Ejecución:

OPERATION	OBJECT_NAME	OPTIONS	CARDINALITY	COST
SELECT STATEMENT			270	144
NESTED LOOPS			270	144
NESTED LOOPS			270	144
NESTED LOOPS			270	144
TABLE ACCESS	RESERVASERVICIOS	FULL	270	144
Filter Predicates				
AND				
RS.FECHAINICIO>=:FECHA1				
RS.FECHAFIN<=:FECHA2				
TABLE ACCESS	OFERTASHABITACIONES	BY INDEX ROWID	1	0
INDEX	OFERTASHABITACIONES_PK	UNIQUE SCAN	1	0
Access Predicates				
RS.HABITACION_ID=OH.ID				
INDEX	USUARIOS_PK	UNIQUE SCAN	1	0
Access Predicates				
U.ID=OH.USUARIOS_ID				
TABLE ACCESS	USUARIOS	BY INDEX ROWID	1	0

- RFC10

Sentencia SQL:

Universidad de Los Andes

Sistemas Transaccionales

Iteración 02: Documento de Análisis

Juliana Sofía Ahumada Arcos – 201921471

Juan Felipe Caraballo Umbarila – 201923741

Sebastián Umaña Peinado – 202013778

```
SELECT u1.id, u1.nombre, u1.apellido FROM usuarios u1
WHERE u1.id NOT IN
(SELECT u.id FROM
RESERVASERVICIOS rs JOIN
OFERTASHABITACIONES oh ON rs.habitacion_id = oh.id
JOIN USUARIOS u
ON u.id = oh.usuarios_id
WHERE rs.fechainicio >= '01-FEB-2023' AND rs.fechafin <= '04-FEB-2023');
```

Plan de Ejecución:

OPERATION	OBJECT_NAME	OPTIONS	CARDINALITY	COST
SELECT STATEMENT				103789
MERGE JOIN		ANTI		103789
TABLE ACCESS	USUARIOS	BY INDEX ROWID		103789
INDEX	USUARIOS_PK	FULL SCAN		103789
SORT		UNIQUE		4
Access Predicates				
U1.ID=ID				
Filter Predicates				
U1.ID=ID				
VIEW	SYS.VW_NSO_1			4
NESTED LOOPS				4
NESTED LOOPS				4
TABLE ACCESS	RESERVASERVICIOS	FULL		4
Filter Predicates				
AND				
RS.FECHAINICIO>='01-FEB-2023'				
RS.FECHAFIN<='04-FEB-2023'				
INDEX	OFERTASHABITACIONES_PK	UNIQUE SCAN		1
Access Predicates				
RS.HABITACION_ID=OH.ID				
TABLE ACCESS	OFERTASHABITACIONES	BY INDEX ROWID		1

- RFC11

Sentencia SQL:

```
--Servicio más consumido por semana:
WITH ServiciosPorSemana AS (
  SELECT
    TO_CHAR(r.fechainicio, 'YYYY-IW') AS semana_del_anio,
    s.nombre AS servicio_mas_consumido,
    COUNT(*) AS cantidad_consumos,
    DENSE_RANK() OVER (PARTITION BY TO_CHAR(r.fechainicio, 'YYYY-IW') ORDER BY COUNT(*) DESC) AS ranking
  FROM
    reservaservicios r
  JOIN
    servicios s ON r.servicios_id = s.id
  GROUP BY
    TO_CHAR(r.fechainicio, 'YYYY-IW'), s.nombre
)
SELECT
  semana_del_anio,
  servicio_mas_consumido,
  cantidad_consumos
FROM
  ServiciosPorSemana
WHERE
  ranking = 1;

--Servicio menos consumido por semana:
WITH ServiciosPorSemana AS (
  SELECT
    TO_CHAR(r.fechainicio, 'YYYY-IW') AS semana_del_anio,
    s.nombre AS servicio_menos_consumido,
    COUNT(*) AS cantidad_consumos,
    DENSE_RANK() OVER (PARTITION BY TO_CHAR(r.fechainicio, 'YYYY-IW') ORDER BY COUNT(*) ASC) AS ranking
  FROM
    reservaservicios r
  JOIN
    servicios s ON r.servicios_id = s.id
  GROUP BY
    TO_CHAR(r.fechainicio, 'YYYY-IW'), s.nombre
)
SELECT
  semana_del_anio,
  servicio_menos_consumido,
  cantidad_consumos
FROM
  ServiciosPorSemana
WHERE
  ranking = 1;
```

Universidad de Los Andes

Sistemas Transaccionales

Iteración 02: Documento de Análisis

Juliana Sofía Ahumada Arcos – 201921471

Juan Felipe Caraballo Umbarila – 201923741

Sebastián Umaña Peinado – 202013778

```
--Habitaciones más solicitadas por semana:
WITH HabitacionesSolicitadas AS (
  SELECT
    TO_CHAR(r.fechainicio, 'YYYY-IW') AS semana_del_anio,
    oh.tipohabitacion AS habitacion,
    COUNT(*) AS cantidad_solicitudes,
    RANK() OVER (PARTITION BY TO_CHAR(r.fechainicio, 'YYYY-IW') ORDER BY COUNT(*) DESC) AS ranking
  FROM
    reservahabitaciones r
  JOIN
    ofertashabitaciones oh ON r.ofertashabitaciones_id = oh.id
  GROUP BY
    TO_CHAR(r.fechainicio, 'YYYY-IW'), oh.tipohabitacion
)
SELECT
  semana_del_anio,
  habitacion AS habitacion_mas_solicitada,
  cantidad_solicitudes
FROM
  HabitacionesSolicitadas
WHERE
  ranking = 1;

--Habitaciones menos solicitadas por semana:
WITH HabitacionesSolicitadas AS (
  SELECT
    TO_CHAR(r.fechainicio, 'YYYY-IW') AS semana_del_anio,
    oh.tipohabitacion AS habitacion,
    COUNT(*) AS cantidad_solicitudes,
    RANK() OVER (PARTITION BY TO_CHAR(r.fechainicio, 'YYYY-IW') ORDER BY COUNT(*) ASC) AS ranking
  FROM
    reservahabitaciones r
  JOIN
    ofertashabitaciones oh ON r.ofertashabitaciones_id = oh.id
  GROUP BY
    TO_CHAR(r.fechainicio, 'YYYY-IW'), oh.tipohabitacion
)
SELECT
  semana_del_anio,
  habitacion AS habitacion_menos_solicitada,
  cantidad_solicitudes
FROM
  HabitacionesSolicitadas
WHERE
  ranking = 1;
```

Plan de Ejecución:

OPERATION	OBJECT_NAME	OPTIONS	CARDINALITY	COST
SELECT STATEMENT				107874 155
VIEW				107874 155
Filter Predicates				
RANKING=1				
WINDOW		SORT PUSHED RANK		107874 155
Filter Predicates				
DENSE_RANK() OVER (PARTITION BY TO_CHAR(INTERNAL_FUNCTION(R.FECHAINICIO),'YYYY-IW') ORDER BY COUNT(*)<=1				
HASH		GROUP BY		107874 155
NESTED LOOPS				107874 147
NESTED LOOPS				107874 147
TABLE ACCESS	RESERVASERVICIOS	FULL		107874 143
INDEX	SERVICIOS_PK	UNIQUE SCAN	1	0
Access Predicates				
R.SERVICIOS_ID=S.ID				
TABLE ACCESS	SERVICIOS	BY INDEX ROWID	1	0
SELECT STATEMENT				107874 155
VIEW				107874 155
Filter Predicates				
RANKING=1				
WINDOW		SORT PUSHED RANK		107874 155
Filter Predicates				
DENSE_RANK() OVER (PARTITION BY TO_CHAR(INTERNAL_FUNCTION(R.FECHAINICIO),'YYYY-IW') ORDER BY COUNT(*)<=1				
HASH		GROUP BY		107874 155
NESTED LOOPS				107874 147
NESTED LOOPS				107874 147
TABLE ACCESS	RESERVASERVICIOS	FULL		107874 143
INDEX	SERVICIOS_PK	UNIQUE SCAN	1	0
Access Predicates				
R.SERVICIOS_ID=S.ID				
TABLE ACCESS	SERVICIOS	BY INDEX ROWID	1	0

Universidad de Los Andes

Sistemas Transaccionales

Iteración 02: Documento de Análisis

Juliana Sofía Ahumada Arcos – 201921471

Juan Felipe Caraballo Umbarila – 201923741

Sebastián Umaña Peinado – 202013778

OPERATION	OBJECT_NAME	OPTIONS	CARDINALITY	COST
SELECT STATEMENT				386924 446
VIEW				386924 446
Filter Predicates RANKING=1				
WINDOW		SORT PUSHED RANK		386924 446
Filter Predicates RANK() OVER (PARTITION BY TO_CHAR(INTERNAL_FUNCTION(R.FECHAINICIO),'YYYY-MM') ORDER BY COUNT(*) DESC) <= 1				
HASH		GROUP BY		386924 446
NESTED LOOPS				386924 423
NESTED LOOPS				386924 423
TABLE ACCESS RESERVAHABITACIONES		FULL		386924 406
INDEX OFERTASHABITACIONES_PK		UNIQUE SCAN	1	0
Access Predicates R.OFERTASHABITACIONES_ID=OH.ID				
TABLE ACCESS OFERTASHABITACIONES		BY INDEX ROWID	1	0

OPERATION	OBJECT_NAME	OPTIONS	CARDINALITY	COST
SELECT STATEMENT				386924 446
VIEW				386924 446
Filter Predicates RANKING=1				
WINDOW		SORT PUSHED RANK		386924 446
Filter Predicates RANK() OVER (PARTITION BY TO_CHAR(INTERNAL_FUNCTION(R.FECHAINICIO),'YYYY-MM') ORDER BY COUNT(*) DESC) <= 1				
HASH		GROUP BY		386924 446
NESTED LOOPS				386924 423
NESTED LOOPS				386924 423
TABLE ACCESS RESERVAHABITACIONES		FULL		386924 406
INDEX OFERTASHABITACIONES_PK		UNIQUE SCAN	1	0
Access Predicates R.OFERTASHABITACIONES_ID=OH.ID				
TABLE ACCESS OFERTASHABITACIONES		BY INDEX ROWID	1	0

Proceso de carga de datos:

En el repositorio del proyecto, subimos una carpeta llamada: cargar_datos, la cual aloja una carpeta para cada respectiva tabla, cada carpeta contiene un código en python que genera un archivo .CSV que contiene los datos para poblar masivamente la base de datos. Ya nosotros generamos los CSV pero pueden generar otros con el fin de probar la integridad y funcionamiento de los mismos archivos.

Cada archivo python usa la librería Faker que me ayuda a generar datos aleatorios para poder dar datos que concuerden con lo estipulado por la estructura de la tabla y usa adicionalmente la librería CSV para escribir sobre archivos CSV y poder introducir los datos generados por la librería Faker.

Luego de generados los CSV nos dirigimos a SQL Developer donde posteriormente se debieron haber creado las tablas con el schema hotel de los andes proporcionado en el repositorio. Al tener todas las tablas creadas seguimos los siguientes pasos:

Ejemplo para la tabla usuarios:

Universidad de Los Andes

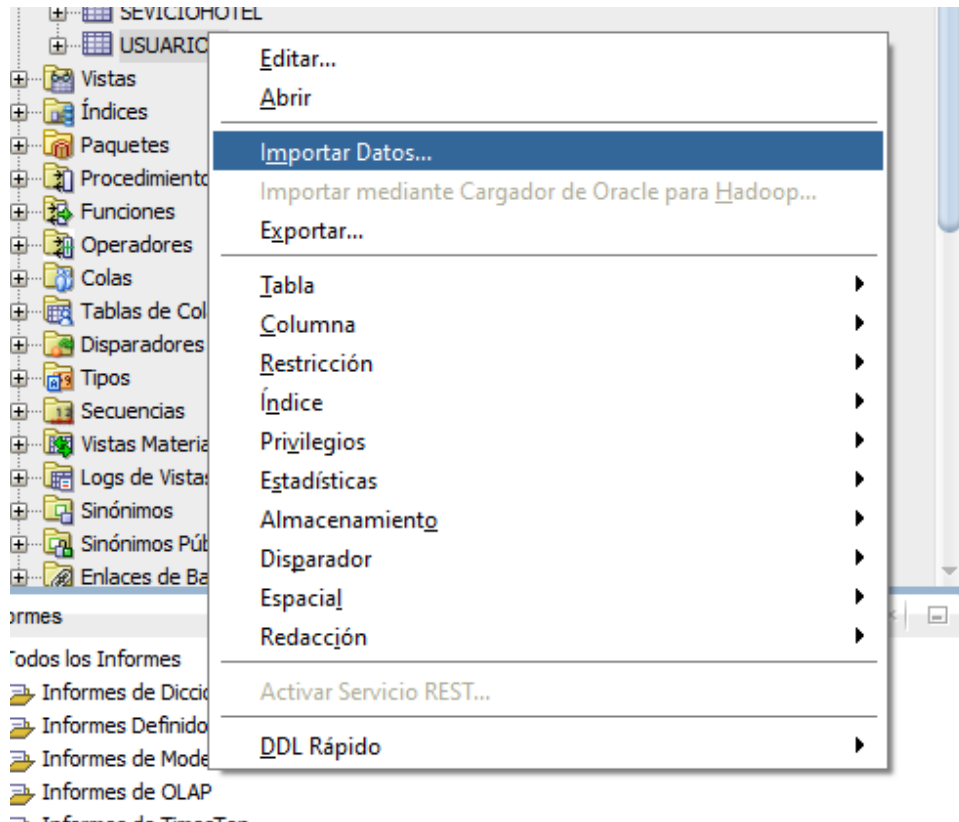
Sistemas Transaccionales

Iteración 02: Documento de Análisis

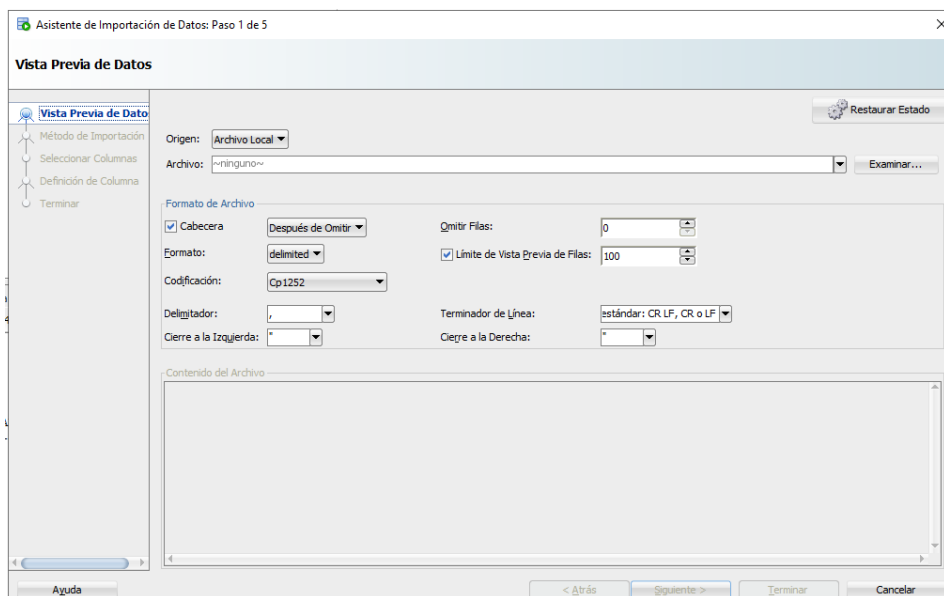
Juliana Sofía Ahumada Arcos – 201921471

Juan Felipe Caraballo Umbarila – 201923741

Sebastián Umaña Peinado – 202013778



Primero damos click derecho e importar datos



Universidad de Los Andes

Sistemas Transaccionales

Iteración 02: Documento de Análisis

Juliana Sofía Ahumada Arcos – 201921471

Juan Felipe Caraballo Umbarila – 201923741

Sebastián Umaña Peinado – 202013778

Ahora examinar y buscamos el CSV correspondiente para la tabla usuarios en el repositorio

Asistente de Importación de Datos: Paso 1 de 5

Vista Previa de Datos

Origen: Archivo Local

Archivo: C:\Users\pebas\OneDrive\Escritorio\cargar_datos\USUARIOS\usuarios_data.csv

Formato de Archivo

☒ Cabecera Después de Omitir

Formato: csv

Codificación: Cp1252

Delimitador: ,

Cierre a la Izquierda: *

Omitir Filas: 0

☒ Límite de Vista Previa de Filas: 100

Terminador de Línea: estándar: CR LF, CR o LF

Cierre a la Derecha: *

Contenido del Archivo

id	nombre	apellido	correo	rol
1	Allen	Sweeney	wbarnes@e...	Gerente
2	Jared	Jimenez	ncampbell@...	Administrador
3	Matthew	Johnson	john87@ex...	Administrador
4	Steven	Jimenez	alexis27@ex...	Administrador
5	Megan	Cruz	frankwilson...	Administrador
6	Laura	Ford	santiagodan...	Cliente
7	David	Williams	linda64@ex...	Cliente
8	Derrick	Williams	brianparker...	Administrador
9	Amanda	Willis	gonzalezjoh...	Gerente
10	Marie	Rasmussen	daltorhanna...	Gerente

Saldrá una vista previa para ver los datos cargados, por lo que, proseguimos dando siguiente

Asistente de Importación de Datos: Paso 2 de 4

Método de Importación

Especifique el método para importar los datos. En el caso del método de tabla externa en área temporal, se creará una tabla externa como tabla intermedia para importar la tabla de destino. Para otros métodos de importación, los datos se importan directamente a la tabla.

Método de Importación: Insertar

☐ Enviar Crear Script a Hoja de Trabajo de SQL

Nombre de la Tabla: USUARIOS

☐ Límite de Importación de Filas: 100

Contenido del Archivo

id	nombre	apellido	correo	rol
1	Allen	Sweeney	wbarnes@e...	Gerente
2	Jared	Jimenez	ncampbell@...	Administrador
3	Matthew	Johnson	john87@ex...	Administrador
4	Steven	Jimenez	alexis27@ex...	Administrador
5	Megan	Cruz	frankwilson...	Administrador
6	Laura	Ford	santiagodan...	Cliente
7	David	Williams	linda64@ex...	Cliente
8	Derrick	Williams	brianparker...	Administrador
9	Amanda	Willis	gonzalezjoh...	Gerente
10	Marie	Rasmussen	daltorhanna...	Gerente
11	Kristy	Vaughan	kristine02@...	Administrador
12	Yvonne	Roach	donnaeole...	Gerente
13	Patrick	Shea	thorntonjul...	Administrador
14	David	Allen	douglasflow...	Administrador
15	Margaret	Horton	tyrone59@e...	Cliente
16	Michael	Stewart	michael17@...	Gerente
17	Daniel	Bovd	oachecoore...	Recepcionista

Dejamos el método de importación "INSERTAR" que hace un insert por cada linea del CSV

Universidad de Los Andes

Sistemas Transaccionales

Iteración 02: Documento de Análisis

Juliana Sofía Ahumada Arcos – 201921471

Juan Felipe Caraballo Umbarila – 201923741

Sebastián Umaña Peinado – 202013778

De aquí en adelante solo es continuar, pulsando “siguiente” para los próximos dos pasos y de no haber problemas de integridad, se van a insertar todas las columnas de manera muy rápida y sin ver los queries.

Cabe aclarar que el orden de llenado de las tablas va en función de la dependencia entre tablas por lo que no puedo llenar una tabla que dependa de otra por restricciones de FK.

Escenario de pruebas:

Para los escenarios de prueba, vamos a usar datos de nuestra base de datos luego de ser cargada masivamente como se menciona en el ítem anterior. Adicionalmente debe correr el programa generando la conexión entre java y oracle mediante el archivo “application.properties”

Cuando el programa se encuentra en ejecución, debe ir a su navegador e ingresar con el siguiente link: <http://localhost:8080/Hoteles/>

No olvide tener el puerto 8080 disponible

Va a observar la siguiente interfaz:

Hotel de Los Andes



Usuarios	Habitaciones	Reserva Servicios	Registro Consumos	Hoteles
Servicios	Planes de Consumo	Ilegadas y salidas	Reserva Habitaciones	
Dinero Recolectado por Servicios Habitaciones		Servicios Más populares		
Porcentaje de ocupación por habitación		fechas con mayor ocupacion		
Buenos Clientes	Servicios con no mucha demanda			
Consultar consumo por usuario en rango de fechas			Consultas Avanzadas	

Universidad de Los Andes

Sistemas Transaccionales

Iteración 02: Documento de Análisis

Juliana Sofía Ahumada Arcos – 201921471

Juan Felipe Caraballo Umbarila – 201923741

Sebastián Umaña Peinado – 202013778

RFC1:

Para acceder al requerimiento 1 debe seleccionar en la interfaz el botón que dice lo siguiente: Dinero recolectado por servicio Habitaciones

Dinero recolectado por reservas de cada habitación en el ultimo año	
Id Habitación	Total Recibido
73627	433.0
76808	227.0
80169	122.0
35619	70.0
88960	356.0
45112	78.0
42645	344.0
90199	400.0
57911	208.0
72605	499.0


Para nuestro esquema, este es el resultado esperado

RFC2:


Para acceder al requerimiento 2 debe seleccionar en la interfaz el botón que dice lo siguiente: Servicios más populares:

Establecer fechas para buscar servicios populares

Fecha Inicio:



Fecha Fin:



BuscarCancelar

Universidad de Los Andes

Sistemas Transaccionales

Iteración 02: Documento de Análisis

Juliana Sofía Ahumada Arcos – 201921471

Juan Felipe Caraballo Umbarila – 201923741

Sebastián Umaña Peinado – 202013778

Para este escenario pondremos las fechas que se indican en la fotografía

Servicios más populares	
Servicio	Cantidad de veces usado
manage	134
energy	131
home	130
left	129
society	129
party	128
budget	128
model	127
yard	126
natural	126
which	125
near	125
evening	125

Para nuestro esquema, este es el resultado esperado

RFC3:

Para acceder al requerimiento 3 debe seleccionar en la interfaz el botón que dice lo siguiente: Porcentaje de ocupación por habitación

% de ocupación de habitaciones en el último año	
Id habitación	% de ocupación en el último año
20682	1%
36376	1%
8697	1%
32139	1%
78105	2%
10128	2%
26881	2%
3772	1%
6073	1%
80764	1%
53574	1%
79559	1%

Para nuestro esquema, este es el resultado esperado

RFC4:

No implementado en la interfaz

Universidad de Los Andes

Sistemas Transaccionales

Iteración 02: Documento de Análisis

Juliana Sofía Ahumada Arcos – 201921471

Juan Felipe Caraballo Umbarila – 201923741

Sebastián Umaña Peinado – 202013778

RFC5:

Para acceder al requerimiento 5 debe seleccionar en la interfaz el botón que dice lo siguiente: Consultar consumo por usuario en rango de fechas

Establecer fechas y id para buscar consumos

Id usuario:

Fecha Inicio:

Fecha Fin:

BuscarCancelar

Usamos al cliente de id 265 y la fechas de la fotografía

Consumos

Nombre Hotel	Fecha Consumo	Descripción	Costo
Bates	2023-10-22 00:00:00.0	Training.	179
Leblanc	2023-11-03 00:00:00.0	Although.	78
Simon	2023-11-06 00:00:00.0	Tree over.	186
Simon	2023-11-04 00:00:00.0	Quickly.	73

Para nuestro esquema, este es el resultado esperado

Universidad de Los Andes

Sistemas Transaccionales

Iteración 02: Documento de Análisis

Juliana Sofía Ahumada Arcos – 201921471

Juan Felipe Caraballo Umbarila – 201923741

Sebastián Umaña Peinado – 202013778

RFC6:

Para acceder al requerimiento 6 debe seleccionar en la interfaz el botón que dice lo siguiente: Fechas con mayor ocupación

Mayor Ocupacion	
Fecha de Inicio	Ocupación
2023-10-15 00:00:00.0	11841
2023-10-29 00:00:00.0	11828
2023-11-03 00:00:00.0	11821
2023-10-30 00:00:00.0	11800
2023-10-25 00:00:00.0	11797
2023-11-02 00:00:00.0	11779
2023-10-22 00:00:00.0	11771
2023-10-19 00:00:00.0	11750
Menor Ocupacion	
Fecha de Inicio	Ocupacion
2023-10-28 00:00:00.0	11502
2023-10-23 00:00:00.0	11508
2023-10-20 00:00:00.0	11526
2023-10-09 00:00:00.0	11559
2023-11-06 00:00:00.0	11562
2023-10-12 00:00:00.0	11569
2023-10-10 00:00:00.0	11575
Mayores Consumos	
Fecha de Inicio	Mayor Consumo
2023-10-15 00:00:00.0	364293
2023-10-20 00:00:00.0	362704
2023-10-30 00:00:00.0	362045
2023-10-16 00:00:00.0	357846
2023-11-03 00:00:00.0	356904
2023-10-29 00:00:00.0	356428
2023-10-25 00:00:00.0	355484
2023-10-19 00:00:00.0	355118

Para nuestro esquema, estos son los resultados esperados

RFC7:

Universidad de Los Andes

Sistemas Transaccionales

Iteración 02: Documento de Análisis

Juliana Sofía Ahumada Arcos – 201921471

Juan Felipe Caraballo Umbarila – 201923741

Sebastián Umaña Peinado – 202013778

Para acceder al requerimiento 7 debe seleccionar en la interfaz el botón que dice lo siguiente: Buenos Clientes

Lista de Buenos Clientes

Usuario ID	Nombre	Apellido	Total Pagos Último Año	Total Días Estadía Último Año
335	Daniel	Underwood	1188	113
426	Jack	Chung	587	48
444	Jeffrey	Thompson	468	51
492	Dale	Smith	942	69
494	Melissa	Martinez	466	42
658	Mitchell	Howard	849	52
765	Adam	Parker	843	87
950	Debbie	Warren	308	41
1256	Lisa	Miller	585	29

Para nuestro esquema, estos son los resultados esperados

RFC8:

Para acceder al requerimiento 8 debe seleccionar en la interfaz el botón que dice lo siguiente: Servicios con no mucha demanda

Servicios Menos Solicitados

ID del Servicio	Nombre del Servicio	Veces Solicitado
Verter Abajo		

Para nuestro esquema, estos son los resultados esperados

No se muestra ningún servicio porque en nuestro esquema, todos los servicios son utilizados, esto se presentó debido a que decidimos tener practicidad en la creación de los datos en el modelo, por lo que todos los servicios son muy usados.

RFC9 y RFC10:

Para acceder al requerimiento 9 y 10 debe seleccionar en la interfaz el botón que dice lo siguiente: Consultas avanzadas

Universidad de Los Andes

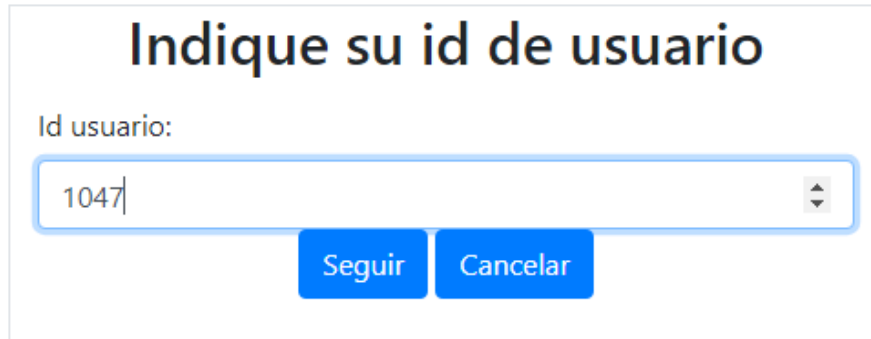
Sistemas Transaccionales

Iteración 02: Documento de Análisis

Juliana Sofía Ahumada Arcos – 201921471

Juan Felipe Caraballo Umbarila – 201923741

Sebastián Umaña Peinado – 202013778



A user login form titled "Indique su id de usuario". It features a text input field with the value "1047" and a dropdown arrow on the right. Below the input field are two blue buttons: "Seguir" and "Cancelar".

Estas 2 consultas requieren de un gerente, un administrador o un recepcionista, en este caso usamos 1047 que es un administrador.

Su rol es Recepcionista

Tiene acceso a las siguientes funciones avanzadas

Consultar consumos por un cliente en el tiempo dado

Consultar clientes que no consumieron X servicio en el tiempo dado

Si su rol es cliente, no puede hacer las consultas del RFC9 y RFC10

Para el RFC9 debe seleccionar: Consultar consumos por un cliente en el tiempo dado

Universidad de Los Andes

Sistemas Transaccionales

Iteración 02: Documento de Análisis

Juliana Sofía Ahumada Arcos – 201921471

Juan Felipe Caraballo Umbarila – 201923741

Sebastián Umaña Peinado – 202013778

Establecer fechas y id para buscar consumos

Id servicio:

20

Fecha Inicio:

01/01/2022

Fecha Fin:

01/01/2024

Buscar

Cancelar

Usamos el id 20 y las fechas de la fotografía

Ordenar por fecha						Ordenar por id del cliente						Ordenar por nombre del cliente						Ordenar por nombre del servicio					
id consumo		Fecha Consumo				Servicio				id Usuario				Nombre									
312		2023-11-01 00:00:00.0				open				25206				Melissa									
313		2023-10-29 00:00:00.0				possible				57948				Sonya									
434		2023-10-29 00:00:00.0				up				58025				Rhonda									
314		2023-11-17 00:00:00.0				music				14917				Joseph									
407		2023-10-30 00:00:00.0				evening				14820				Johnny									
357		2023-10-14 00:00:00.0				never				58199				Tommy									

Para nuestro esquema, estos son los resultados esperados

Para el RFC10 debe seleccionar: Consultar clientes que no consumieron X servicio en el tiempo dado

Universidad de Los Andes

Sistemas Transaccionales

Iteración 02: Documento de Análisis

Juliana Sofía Ahumada Arcos – 201921471

Juan Felipe Caraballo Umbarila – 201923741

Sebastián Umaña Peinado – 202013778

Establecer fechas y id para buscar consumos

Id servicio:

Fecha Inicio:

Fecha Fin:

BuscarCancelar

Usamos el id 20 y las fechas de la fotografía

Ordenar por nombre del cliente			Ordenar por id del cliente		
Id Usuario	Nombre		Apellido		
1	Allen		Sweeney		
2	Jared		Jimenez		
3	Matthew		Johnson		
4	Steven		Jimenez		
5	Megan		Cruz		
6	Laura		Ford		
7	David		Williams		

Para nuestro esquema, estos son los resultados esperados

Universidad de Los Andes

Sistemas Transaccionales

Iteración 02: Documento de Análisis

Juliana Sofía Ahumada Arcos – 201921471

Juan Felipe Caraballo Umbarila – 201923741

Sebastián Umaña Peinado – 202013778

RFC11:

Para acceder al requerimiento 11 debe seleccionar en la interfaz el botón que dice lo siguiente: Consultas avanzadas

Indique su id de usuario

Id usuario:

SeguirCancelar

Debe usar el id 262 que corresponde a un gerente general

Para el RFC10 debe seleccionar: Consultar funcionamiento

Rankings Semanales

Servicio más consumido

# de semana	Servicio más consumido	Cantidad de veces
2023-40	recent	13
2023-41	recent	74
2023-42	recent	68
2023-43	note	74
2023-44	radio	75
2023-45	suffer	14

Servicio menos consumido

# de semana	Servicio menos consumido	Cantidad de veces
2023-40	either	1
2023-40	white	1
2023-40	poor	1
2023-40	perform	1
2023-40	mission	1
2023-40	behind	1
2023-40	institution	1
2023-40	seek	1
2023-40	face	1

Universidad de Los Andes

Sistemas Transaccionales

Iteración 02: Documento de Análisis

Juliana Sofía Ahumada Arcos – 201921471

Juan Felipe Caraballo Umbarila – 201923741

Sebastián Umaña Peinado – 202013778

Habitación más solicitada

# de semana	Habitación más solicitada	Cantidad de veces
2023-40	Individual	3953
2023-41	Individual	27298
2023-42	Doble	27367
2023-43	Individual	27244
2023-44	Individual	27625
2023-45	Doble	3891

Habitación menos solicitada

# de semana	Habitación menos solicitada	Cantidad de veces
2023-40	Doble	3803
2023-41	Doble	26956
2023-42	Individual	27036
2023-43	Suite	27096
2023-44	Suite	27130
2023-45	Individual	3814

Para nuestro esquema, estos son los resultados esperados

RFC12:

No se encuentra implementado en la interfaz