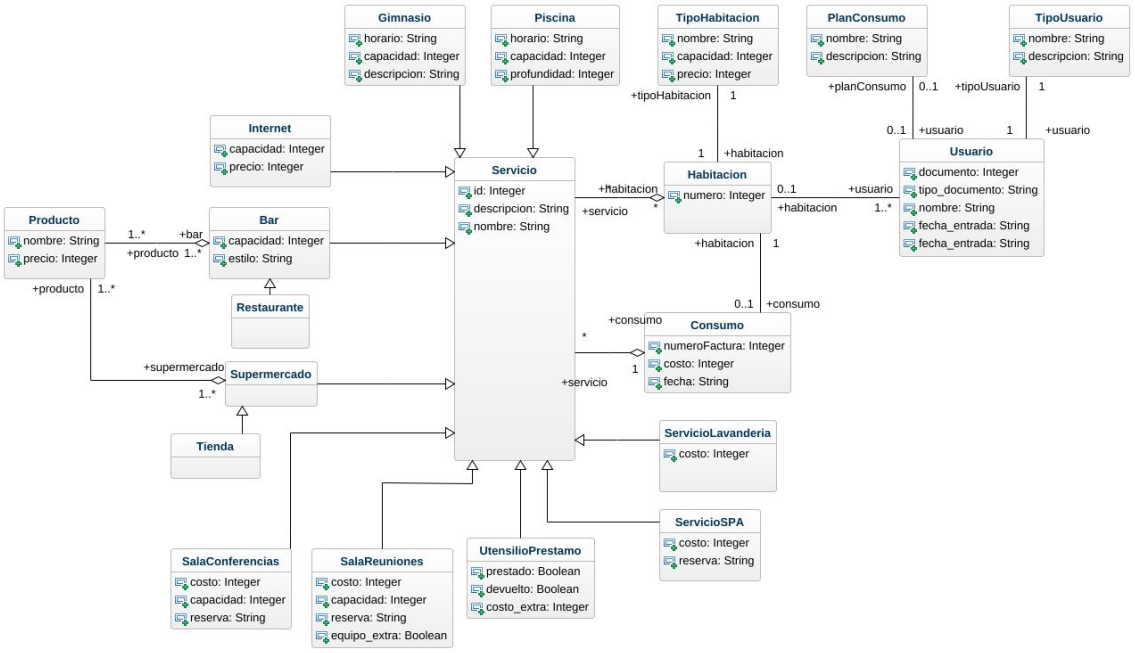
Entrega 2 Proyecto – Hotel de los Andes

Documento de informe y diseño

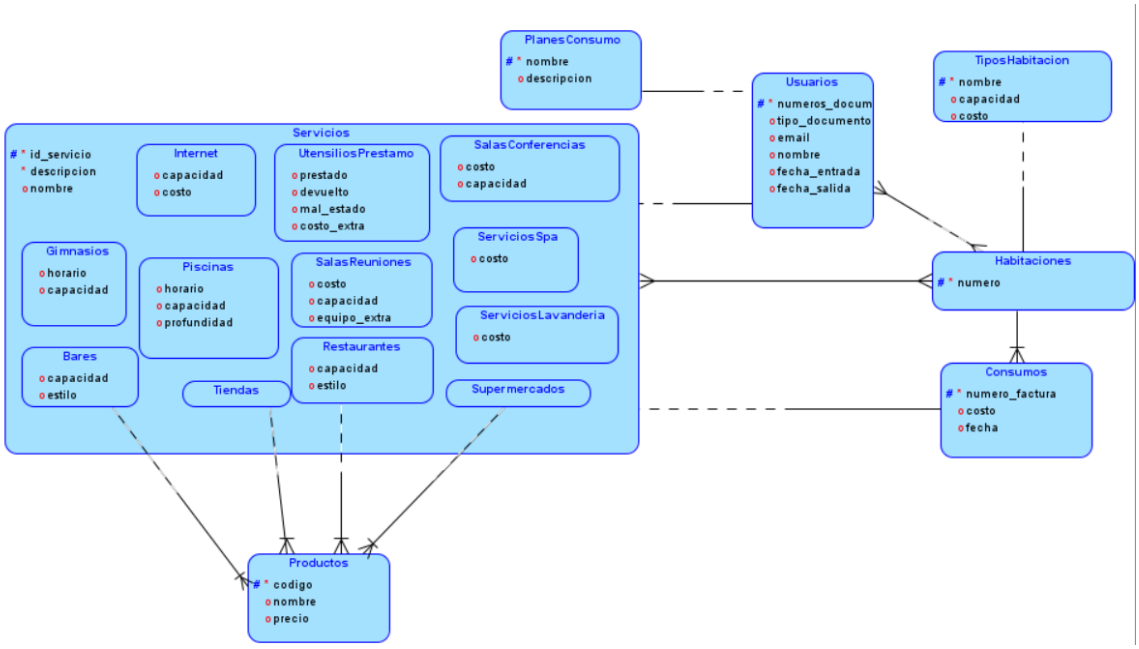
- Julián David Bohórquez Beltrán - 202121973
- Santiago Rodríguez Mora - 202110332
- Valeria Torres Gómez – 202110363

Análisis

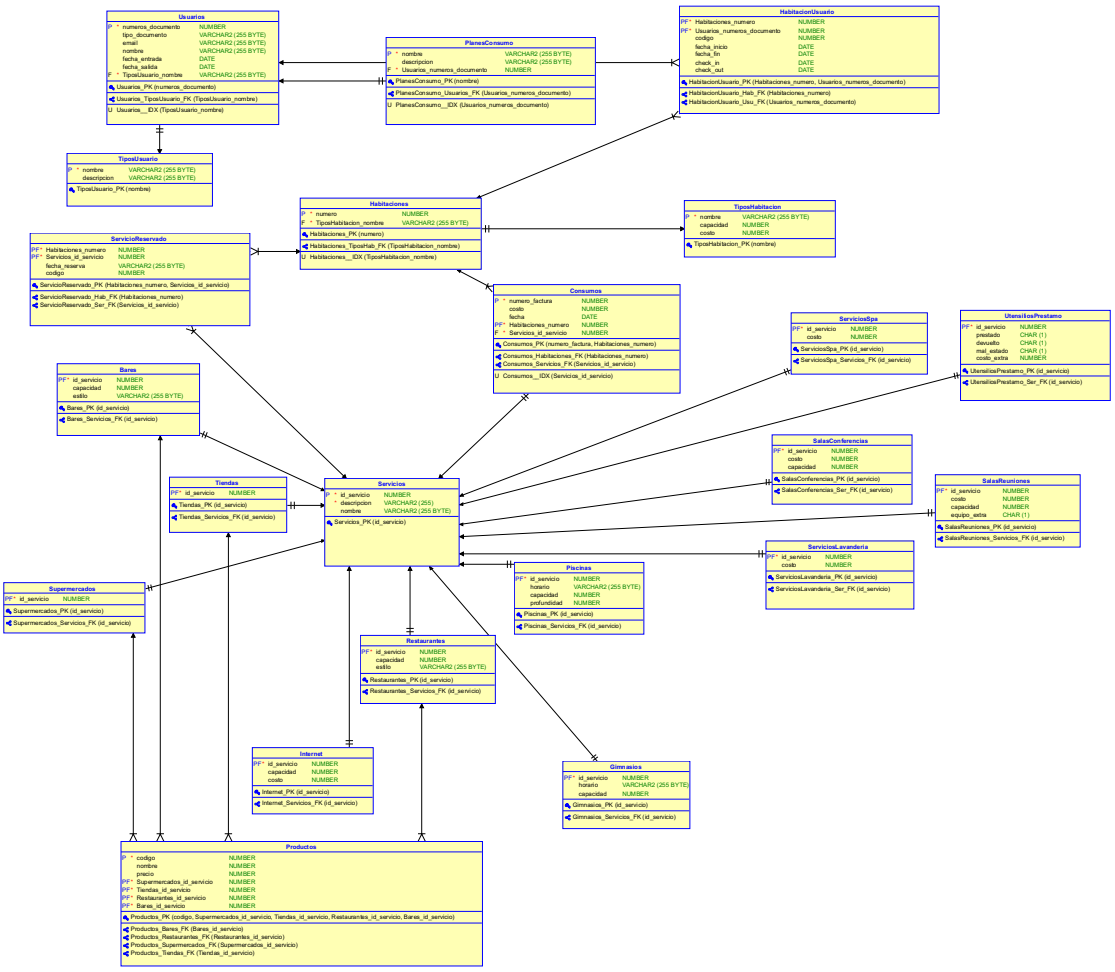
Para los ajustes del modelo conceptual se hizo el ajuste respecto a las clases heredadas de usuario, ya que al no tener nuevos atributos se volvía redundante crear una herencia desde dicha clase a otras, como por ejemplo gerente, administrador, empleado, etc. El modelo consolidado quedo de la siguiente manera:



Luego se realizó el ajuste correspondiente a los cambios anteriores en el modelo relacional:



Además, se realizó el ajuste correspondiente en el modelo entidad-relación:



Diseño de la aplicación

Índices:

- RFC1 - MOSTRAR EL DINERO RECOLECTADO POR SERVICIOS EN CADA HABITACIÓN EN EL ÚLTIMO AÑO CORRIDO:

  No es necesario un índice ya que los datos se consultan sobre un rango de tiempo específico y se calculan agregados por habitación. La selectividad es baja ya que se recopila información de todas las habitaciones durante un año.

- RFC2 - MOSTRAR LOS 20 SERVICIOS MÁS POPULARES:

  Sí es conveniente crear un índice sobre la popularidad de los servicios ya que sería útil para acelerar la consulta. La selectividad puede variar, pero si se actualiza con regularidad, un índice secundario sería apropiado.

- RFC3 - MOSTRAR EL ÍNDICE DE OCUPACIÓN DE CADA UNA DE LAS HABITACIONES DEL HOTEL:

  Sería conveniente crear un índice para el porcentaje de ocupación de las habitaciones. La consulta se realiza sobre un período de tiempo y aunque puede ser actualizado con frecuencia, ayudaría a acelerar las consultas.

- RFC4 - MOSTRAR LOS SERVICIOS QUE CUMPLEN CON CIERTA CARACTERÍSTICA:

  Sí sería conveniente un índice compuesto por las características relevantes sería útil. La selectividad depende de las características seleccionadas, pero un índice secundario sería adecuado.

- RFC5 - MOSTRAR EL CONSUMO EN HOTELANDES POR UN USUARIO DADO, EN UN RANGO DE FECHAS INDICADO:

  Sí se necesita un índice en el nombre del usuario y las fechas de consumo. La selectividad puede variar, pero un índice secundario sería útil.

- RFC6 - ANALIZAR LA OPERACIÓN DE HOTELANDES:

  No es necesario crear un índice ay que las fechas de mayor ocupación, ingresos y menor demanda se calculan una vez y se muestran como informes. No se realizan búsquedas en tiempo real.

- RFC7 - ENCONTRAR LOS BUENOS CLIENTES:

  Un índice en el gasto y la frecuencia de estancia sería útil. La selectividad depende de los criterios, pero un índice secundario es adecuado.

- RFC8 - ENCONTRAR LOS SERVICIOS QUE NO TIENEN MUCHA DEMANDA:

  No es necesario un índice. La consulta busca servicios con baja demanda, y no es común buscarlos en tiempo real. Con la actualización periódica es suficiente.

- RFC9 - CONSULTAR CONSUMO EN HOTELANDES:

  Un índice en el servicio y las fechas de consumo es útil. La selectividad depende del servicio y la fecha, pero un índice secundario es apropiado.

- RFC10 - CONSULTAR CONSUMO EN HOTELANDES – RFC9-V2

  Un índice en el servicio y las fechas de consumo es útil. La selectividad depende del servicio y la fecha, pero un índice secundario es apropiado.

- RFC11 - CONSULTAR FUNCIONAMIENTO:

  No es necesario un índice. La consulta se basa en generar informes sobre el funcionamiento general del hotel, y no se buscan datos en tiempo real.

- RFC12 - CONSULTAR LOS CLIENTES EXCELENTES:

  Un índice en la frecuencia de estancia y el gasto de los clientes sería útil. La selectividad dependerá de los tres tipos de clientes excelentes, y un índice secundario es apropiado.

Índices creados por Oracle:

CONSUMOS_IDX

	INDEX_OWNER	INDEX_NAME	TABLE_OWNER	TABLE_NAME	COLUMN_NAME	COLUMN_POSITION	DESCEND
1	ISIS2304B31202320	CONSUMOS__IDX	ISIS2304B31202320	CONSUMOS	SERVICIOS_ID_SERVICIO	1	ASC

un índice es una estructura que se crea para acelerar la recuperación y ordenamiento de datos en una tabla. Oracle utiliza índices para mejorar el rendimiento de las consultas al permitir un acceso más rápido a los registros en una tabla. Por lo mismo se ha creado un índice en la tabla "consumos" en la columna "servicios_id_servicio". Este índice almacena una lista de valores de la columna y su posición en la tabla. Además, se ha indicado que el índice se utiliza para ordenar los datos en esta columna de manera descendente. Esto significa que cuando se realiza una consulta que implica ordenar los registros de esa columna en orden descendente, el índice facilitará esta operación de manera eficiente.

PLANESCONSUMO_IDX

	INDEX_OWNER	INDEX_NAME	TABLE_OWNER	TABLE_NAME	COLUMN_NAME	COLUMN_POSITION	DESCEND
1	ISIS2304B31202320	PLANESCONSUMO__IDX	ISIS2304B31202320	PLANESCONSUMO	USUARIOS_NUMEROS_DOCUMENTO	1	ASC

Ahora en la tabla "planesconsumo" de la base de datos Oracle, se ha creado un índice en la columna "usuarios_numeros_documento" para acelerar las búsquedas y el ordenamiento de datos en esta columna. Este índice permitirá tanto

ordenamiento ascendente como descendente, mejorando así el rendimiento de las consultas relacionadas con esta columna.

Diseño de las consultas:

```
--RFC1 - MOSTRAR EL DINERO RECOLECTADO POR SERVICIOS EN CADA HABITACIÓN EN EL
ÚLTIMO AÑO CORRIDO

Select h.numero, co.costo
FROM Consumos Co, Habitaciones H, Servicios S
WHERE co.habitaciones_numero = H.numero
AND co.servicios_id_servicio = s.id_servicio;
```

```
--RFC2 - MOSTRAR LOS 20 SERVICIOS MÁS POPULARES
Select s.nombre, Count(nombre) as popularidad
FROM servicios s, Consumos c
WHERE c.servicios_id_servicio = s.id_servicio AND ROWNUM <= 20
GROUP BY s.nombre
ORDER BY popularidad DESC;
```

```
--RFC3 - MOSTRAR EL ÍNDICE DE OCUPACIÓN DE CADA UNA DE LAS HABITACIONES DEL HOTEL
SELECT
    H.numero,
    (SUM(TO_DATE(Fecha_Fin, 'YYYY-MM-DD') - TO_DATE(Fecha_Inicio, 'YYYY-MM-DD'))
* 100.0 / 365) AS IndiceOcupacion
FROM
    HabitacionUsuario HU
JOIN
    Habitaciones H ON hu.habitaciones_numero = H.numero
WHERE
    (TO_DATE(Fecha_Fin, 'YYYY-MM-DD') >= (TO_DATE(Fecha_Inicio, 'YYYY-MM-DD')))
GROUP BY
    H.numero;
```

```
--RFC4 - MOSTRAR LOS SERVICIOS QUE CUMPLEN CON CIERTA CARACTERÍSTICA
SELECT
    s.id_servicio,
    s.descripcion AS descripcion_servicio,
    s.nombre AS nombre_servicio,
    c.fecha AS fecha_consumo,
    c.costo AS costo_consumo,
    e.nombre AS nombre_empleado,
    t.nombre AS tipo_servicio,
    c.categoria AS categoria_servicio
FROM servicios s
INNER JOIN consumos c ON s.id_servicio = c.servicios_id_servicio
```

```sql
LEFT JOIN empleados e ON c.empleado_id = e.id_empleado
INNER JOIN tipos_servicio t ON s.tipo_servicio_id = t.id_tipo_servicio
WHERE
    c.costo BETWEEN 50 AND 100
    AND c.fecha BETWEEN '2022-01-01' AND '2023-01-01'
    AND e.nombre = 'Nombre del Empleado'
    AND t.nombre = 'Tipo de Servicio'
    AND s.categoria = 'Categoría del Servicio';
```

```sql
--RFC5 - MOSTRAR EL CONSUMO EN HOTELANDES POR UN USUARIO DADO, EN UN RANGO DE
FECHAS INDICADO
Select u.nombre ,hu.usuarios_numeros_documento, cu.costo
from HabitacionUsuario HU, Usuarios U, Habitaciones H, Consumos CU
WHERE hu.usuarios_numeros_documento = u.numero_documento
AND hu.habitaciones_numero = h.numero
AND h.numero = cu.Habitaciones_numero
AND "?" >= hu.fecha_inicio AND "?" <= hu.fecha_fin;
```

```sql
--RFC6 - ANALIZAR LA OPERACIÓN DE HOTELANDES
    --Mayor concurrencia
SELECT Fecha, CantidadUsuarios
FROM (
    SELECT Fecha, COUNT(*) AS CantidadUsuarios
    FROM (
        SELECT DISTINCT HU.Fecha_Inicio AS Fecha FROM HabitacionUsuario HU
        UNION ALL
        SELECT DISTINCT HU.Fecha_fin AS Fecha FROM HabitacionUsuario HU
    ) Fechas
    GROUP BY Fecha
    ORDER BY CantidadUsuarios DESC
)
WHERE ROWNUM <= 5;
    --Menor concurrencia
SELECT Fecha, CantidadUsuarios
FROM (
    SELECT Fecha, COUNT(*) AS CantidadUsuarios
    FROM (
        SELECT DISTINCT HU.Fecha_Inicio AS Fecha FROM HabitacionUsuario HU
        UNION ALL
        SELECT DISTINCT HU.Fecha_fin AS Fecha FROM HabitacionUsuario HU
    ) Fechas
    GROUP BY Fecha
    ORDER BY CantidadUsuarios ASC
)
WHERE ROWNUM <= 5;
```

```sql
--RFC7 - ENCONTRAR LOS BUENOS CLIENTES
Select u.nombre, hu.usuarios_numeros_documento, hu.fecha_inicio, hu.fecha_fin
```

```sql
FROM HabitacionUsuario hu, Usuarios u
WHERE hu.usuarios_numeros_documento = u.numero_documento
AND (TO_DATE(HU.Fecha_fin, 'YYYY-MM-DD') - TO_DATE(HU.Fecha_Inicio, 'YYYY-MM-
DD')) >= 14
ORDER BY u.nombre;
```

```sql
--RFC8 - ENCONTRAR LOS SERVICIOS QUE NO TIENEN MUCHA DEMANDA
Select s.nombre, COUNT(SR.servicios_id_servicio) AS CantidadDemandas
FROM ServicioReservado SR, Servicios S
WHERE sr.servicios_id_servicio = s.id_servicio
GROUP BY S.nombre
Order by CantidadDemandas asc;
```

```sql
--RFC9 - CONSULTAR CONSUMO EN HOTELANDES
SELECT u.nombre, s.nombre
FROM HabitacionUsuario hu, Usuarios u, habitaciones h, consumos co, servicios s
WHERE hu.usuarios_numeros_documento = u.numero_documento
AND hu.habitaciones_numero = h.numero
AND h.numero = co.habitaciones_numero
AND co.servicios_id_servicio = s.id_servicio
AND s.id_servicio = 12
ORDER BY u.nombre ASC;
```

```sql
--RFC10 - CONSULTAR CONSUMO EN HOTELANDES - RFC9-V2
SELECT u.nombre, consul.nombre
FROM HabitacionUsuario hu, Usuarios u, habitaciones h,
(
SELECT co.habitaciones_numero, s.id_servicio, s.nombre
FROM Consumos co, Servicios s
Where co.servicios_id_servicio = s.id_servicio
) consul
WHERE hu.usuarios_numeros_documento = u.numero_documento
AND h.numero = hu.habitaciones_numero
AND consul.habitaciones_numero = h.numero
AND consul.id_servicio != 10;
```

```sql
--RFC11 - CONSULTAR FUNCIONAMIENTO
WITH Semanas AS (
  SELECT
    TO_CHAR(fecha, 'IYYY-IW') AS semana,
    TO_DATE(TO_CHAR(fecha, 'IYYY-IW') || '6', 'IYYY-IWd') AS inicio_semana,
    TO_DATE(TO_CHAR(fecha, 'IYYY-IW') || '12', 'IYYY-IWd') AS fin_semana
  FROM consumos
)
SELECT
  s.semana AS semana_del_anio,
  (
    SELECT descripcion
```

```
    FROM servicios
    WHERE id_servicio = (
      SELECT servicios_id_servicio
      FROM (
        SELECT servicios_id_servicio, COUNT(*) AS cantidad
        FROM consumos c
        WHERE c.fecha BETWEEN s.inicio_semana AND s.fin_semana
        GROUP BY servicios_id_servicio
        ORDER BY cantidad DESC
      ) WHERE ROWNUM = 1
    )
  ) AS servicio_mas_consumido,
  (
    SELECT descripcion
    FROM servicios
    WHERE id_servicio = (
      SELECT servicios_id_servicio
      FROM (
        SELECT servicios_id_servicio, COUNT(*) AS cantidad
        FROM consumos c
        WHERE c.fecha BETWEEN s.inicio_semana AND s.fin_semana
        GROUP BY servicios_id_servicio
        ORDER BY cantidad ASC
      ) WHERE ROWNUM = 1
    )
  ) AS servicio_menos_consumido,
  (
    SELECT TO_CHAR(habitaciones_numero)
    FROM (
      SELECT habitaciones_numero, COUNT(*) AS cantidad
      FROM servicioreservado sr
      WHERE sr.fecha_reserva BETWEEN s.inicio_semana AND s.fin_semana
      GROUP BY habitaciones_numero
      ORDER BY cantidad DESC
    ) WHERE ROWNUM = 1
  ) AS habitacion_mas_solicitada,
  (
    SELECT TO_CHAR(habitaciones_numero)
    FROM (
      SELECT habitaciones_numero, COUNT(*) AS cantidad
      FROM servicioreservado sr
      WHERE sr.fecha_reserva BETWEEN s.inicio_semana AND s.fin_semana
      GROUP BY habitaciones_numero
      ORDER BY cantidad ASC
    ) WHERE ROWNUM = 1
  ) AS habitacion_menos_solicitada
FROM Semanas s
GROUP BY s.semana, s.inicio_semana, s.fin_semana
```

```
ORDER BY s.semana;
```

```
--RFC12 - CONSULTAR LOS CLIENTES EXCELENTES
SELECT
    u.numeros_documento AS numero_documento,
    u.nombre AS nombre_cliente,
    u.email AS correo_cliente,
    u.fecha_entrada AS fecha_entrada,
    u.fecha_salida AS fecha_salida,
    SUM(CASE
        WHEN c.fecha >= TO_DATE(TO_CHAR(TRUNC(u.fecha_entrada, 'Q'), 'YYYY-MM-
DD'), 'YYYY-MM-DD')
             AND c.fecha < TO_DATE(TO_CHAR(ADD_MONTHS(TRUNC(u.fecha_entrada,
'Q'), 3), 'YYYY-MM-DD'), 'YYYY-MM-DD')
        THEN 1
        ELSE 0
    END) AS estancias_trimestrales,
    MAX(s.precio) AS max_precio_consumido,
    MAX(CASE
        WHEN s.precio > 300000
        THEN 1
        ELSE 0
    END) AS consumo_costoso,
    MAX(CASE
        WHEN (sr.fecha_reserva IS NOT NULL AND sr.codigo IS NOT NULL
              AND (s.descripcion = 'SPA' OR s.descripcion = 'Salones de
Reuniones')
              AND (sr.check_out - sr.check_in) > 4)
        THEN 1
        ELSE 0
    END) AS consumo_spa_o_salones
FROM usuarios u
LEFT JOIN consumos c ON u.numeros_documento = c.usuarios_numeros_documento
LEFT JOIN servicios s ON c.servicios_id_servicio = s.id_servicio
LEFT JOIN servicioreservado sr ON u.numeros_documento =
sr.usuarios_numeros_documento
WHERE
    (SUM(CASE
        WHEN c.fecha >= TO_DATE(TO_CHAR(TRUNC(u.fecha_entrada, 'Q'), 'YYYY-MM-
DD'), 'YYYY-MM-DD')
             AND c.fecha < TO_DATE(TO_CHAR(ADD_MONTHS(TRUNC(u.fecha_entrada,
'Q'), 3), 'YYYY-MM-DD'), 'YYYY-MM-DD')
        THEN 1
        ELSE 0
    END) > 0
    OR MAX(s.precio) > 300000
    OR MAX(CASE
        WHEN (sr.fecha_reserva IS NOT NULL AND sr.codigo IS NOT NULL
```

```
             AND (s.descripcion = 'SPA' OR s.descripcion = 'Salones de
Reuniones')
             AND (sr.check_out - sr.check_in) > 4)
        THEN 1
        ELSE 0
    END) = 1)
GROUP BY
    u.numeros_documento,
    u.nombre,
    u.email,
    u.fecha_entrada,
    u.fecha_salida
HAVING
    (SUM(CASE
        WHEN c.fecha >= TO_DATE(TO_CHAR(TRUNC(u.fecha_entrada, 'Q'), 'YYYY-MM-
DD'), 'YYYY-MM-DD')
             AND c.fecha < TO_DATE(TO_CHAR(ADD_MONTHS(TRUNC(u.fecha_entrada,
'Q'), 3), 'YYYY-MM-DD'), 'YYYY-MM-DD')
        THEN 1
        ELSE 0
    END) > 0
    OR MAX(s.precio) > 300000
    OR MAX(CASE
        WHEN (sr.fecha_reserva IS NOT NULL AND sr.codigo IS NOT NULL
             AND (s.descripcion = 'SPA' OR s.descripcion = 'Salones de
Reuniones')
             AND (sr.check_out - sr.check_in) > 4)
        THEN 1
        ELSE 0
    END) = 1)
ORDER BY u.numeros_documento;
```

Diseño y carga masiva de datos

Diseño de datos:

El proceso de diseño y carga masiva de datos en una base de datos es fundamental para garantizar la integridad y la eficiencia de la información almacenada. En este caso, se trabajó con varias tablas, como "bares," "consumos," "habitaciones," "habitacionusuario," "internet," "piscinas," "planescosumo," "servicioreservado," "servicios," "tiposhabitacion," "tipousuario," y "usuarios." A continuación, se describe el proceso detalladamente:

1. Diseño Inicial:

Antes de cargar los datos, es esencial realizar un diseño inicial de la base de datos. Esto implica definir las tablas, sus columnas y las relaciones entre ellas. En el diseño, es necesario determinar cuáles son las tablas que no tienen llaves foráneas o dependencias y cuáles sí las tienen. Las tablas sin llaves foráneas, generalmente, son las tablas principales que almacenan información clave.

2. Carga de Datos:

Una vez que se tiene el diseño, se procede a la carga de datos. Es importante cargar primero las tablas sin llaves foráneas, ya que no dependen de datos de otras tablas. Esto asegura que los datos estén disponibles y que no haya conflictos de integridad referencial.

3. Cumplir con el Tipo de Datos:

Durante la carga de datos, es fundamental asegurarse de que los valores sean coherentes con los tipos de datos establecidos en la creación de las tablas. Por ejemplo, si una columna está definida como "varchar," los datos deben ser cadenas de texto. Si es "date," los datos deben ser fechas válidas.

4. Mantener Integridad Referencial:

Para las tablas que tienen llaves foráneas, como "habitacionusuario," es crucial cargar primero los datos en las tablas de las que dependen. Por ejemplo, antes de cargar datos en "habitacionusuario," debes asegurarte de que las habitaciones y los usuarios estén disponibles y tengan sus llaves primarias definidas.

5. Llaves Primarias y Relaciones:

Asegurarse de que las llaves primarias se definan correctamente es esencial. Las llaves primarias son únicas para cada fila y garantizan la unicidad de los registros. Además, las relaciones entre tablas deben establecerse correctamente, siguiendo las llaves foráneas, para mantener la integridad referencial.

6. Carga Masiva:

Cuando se trata de una carga masiva de datos, es aconsejable utilizar herramientas y scripts especializados que permitan una carga eficiente y rápida. Esto puede incluir el uso de lote de inserciones o herramientas de importación de datos proporcionadas por el sistema de gestión de bases de datos.

7. Verificación y Validación:

Después de cargar los datos, se debe realizar una verificación y validación exhaustiva para asegurarse de que todo esté en orden. Esto incluye comprobar que las relaciones entre las tablas funcionen correctamente y que los datos cumplan con las restricciones y reglas definidas.

Construcción de la aplicación, ejecución de pruebas y análisis de resultados

Las pruebas de los requerimientos funcionales corren correctamente todas en menos de 0.8 segundos como pide el requerimiento no funcional 1 (RNF1). Cargan las consultas solicitadas en Oracle SQL developer, con los siguientes tiempos de ejecución:

RFC1) 0.016 s
RFC2) 0.012 s
RFC3) 0.016 s
RFC4) 0.015 s
RFC5) 0.03 s
RFC6) 0.062 s
RFC7) 0.051 s
RFC8) 0.039 s
RFC9) 0.043 s
RFC10) 0.012 s
RFC11) 0.035 s
RFC12) 0.041 s

Las pruebas de los requerimientos funcionales demuestran un rendimiento óptimo, ya que todas las consultas se ejecutan en menos de 0.8 segundos, cumpliendo así con el requisito no funcional 1 (RNF1). Al cargar las consultas solicitadas en Oracle SQL Developer, los tiempos de ejecución se distribuyen de la siguiente manera:

Las RFC 1 a 8 tienen tiempos de ejecución extremadamente bajos, ya que son consultas básicas en las que la complejidad se centra principalmente en la comparación de tablas. Esto se debe en parte a la eficaz optimización del motor de Oracle, que simplifica la búsqueda y el análisis de datos.

Por otro lado, los RFC 9 a 12 involucran consultas más complejas, lo que naturalmente conlleva un aumento en el tiempo de ejecución. Sin embargo, este incremento no es significativo, dado que las consultas se han diseñado de manera

precisa y eficiente para minimizar la complejidad y optimizar el rendimiento, lo que garantiza una experiencia satisfactoria para los usuarios.