

Documento de informe

Santiago Tenjo - 202113965

Juan Ortega – 202113442

Jose Martinez – 202116677

[(5 %) Análisis 2](#_Toc2123294391)

[(30% ) Diseño de la aplicación: 3](#_Toc26069450)

[(12%) Índices 3](#_Toc1954538666)

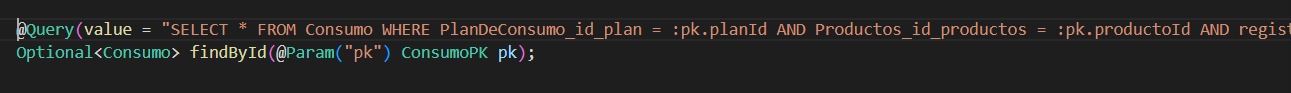
[(18%) Diseño de las consultas: para cada requerimiento funcional de consulta solicitado, documente: 7](#_Toc440065715)

[(25%) Diseño y cargue masivo de datos 7](#_Toc1688785236)

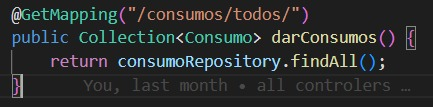
## (5 %) Análisis

Correcciones:

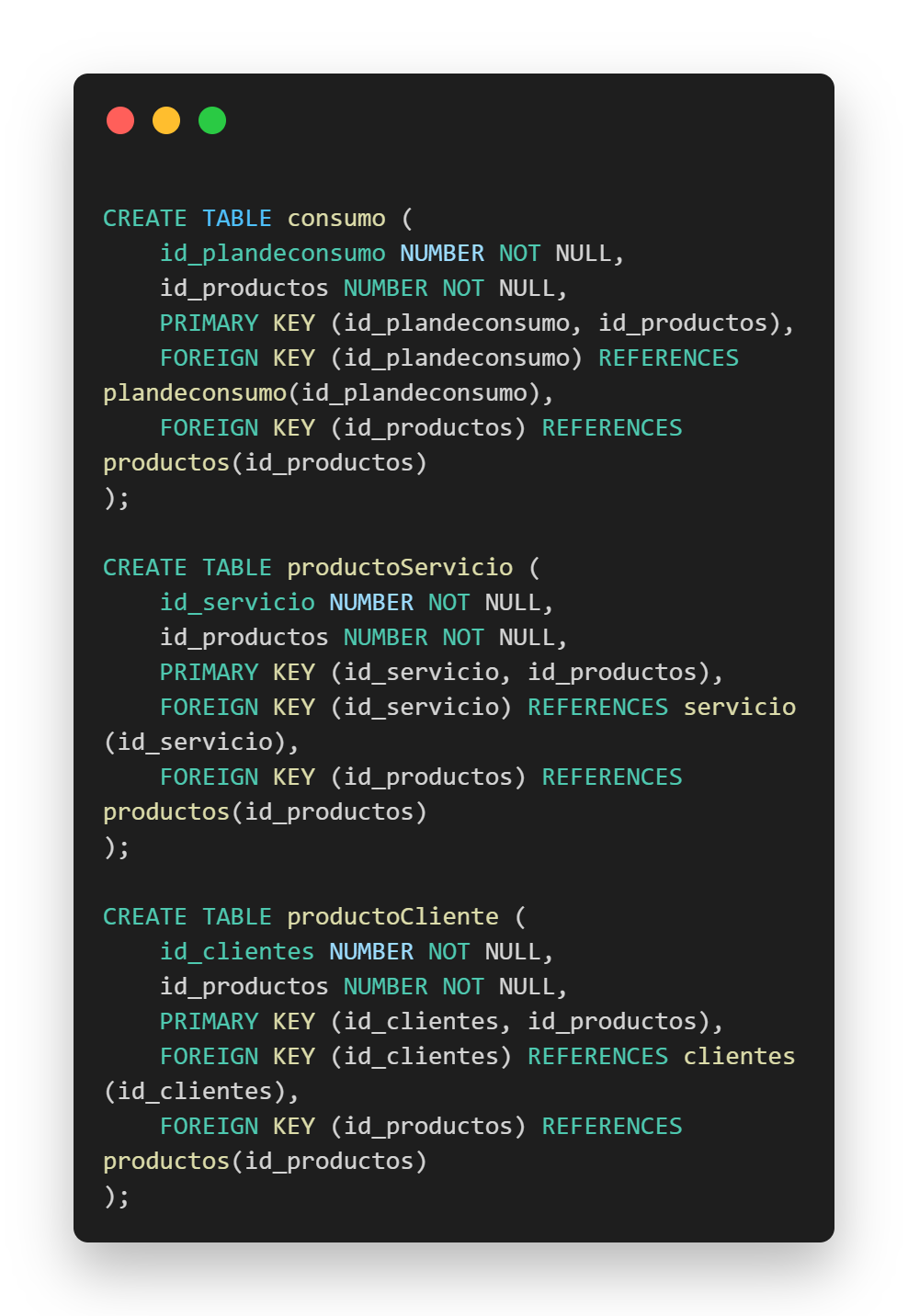
* Clases con fallo se corrigieron a través de un modelo de llaves compuestas



* se cambió la ruta para obtener todos los objetos de una clase



* Se actualizó la creación de tablas, dado que nos habia faltado la creacion de Many To Many



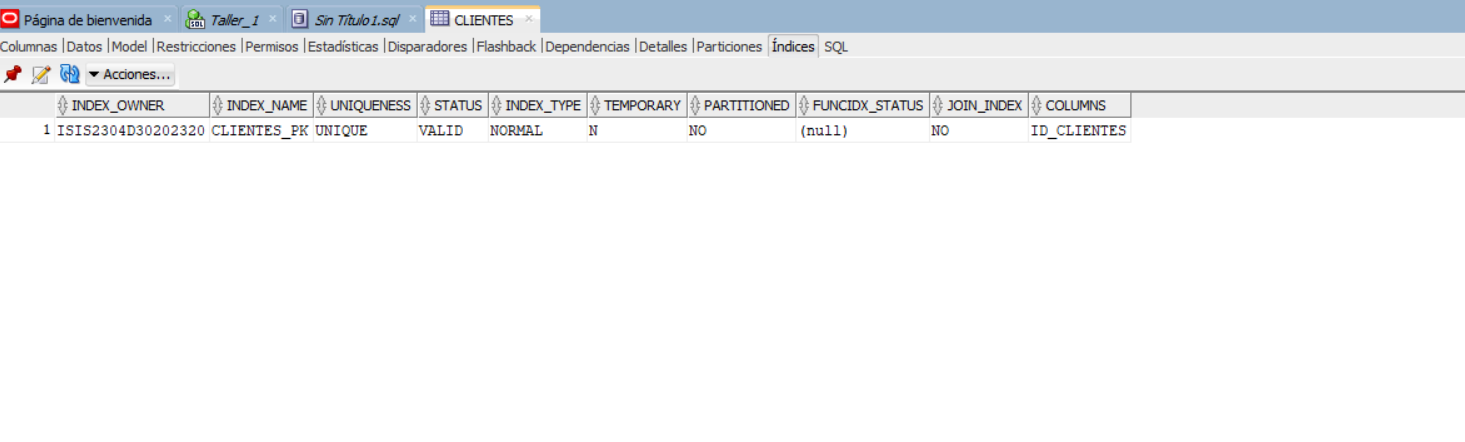
## (30% ) Diseño de la aplicación:

### (12%) Índices

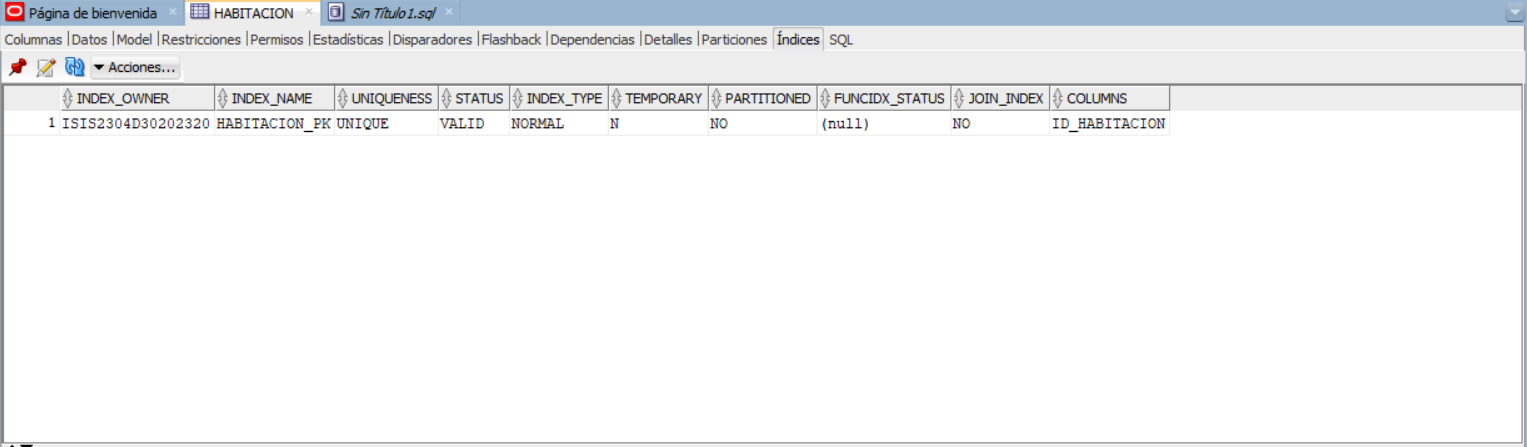
Diseñen los índices necesarios para el adecuado rendimiento global de la aplicación.

**RF1, RF2, RF3, RF4, RF5 y RF6: carga de las tablas con la información de base utilizando sentencias SQL INSERT. La información de base corresponde a los usuarios y la descripción del hotel.**

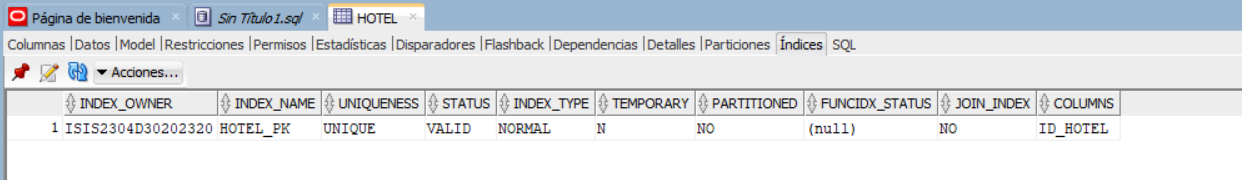
* Es importante que la carga de datos cuente con índices debido a que acelera la inserción de datos para tablas grandes con consultas concurrentes.
* El índice depende de las consultas de selección que se realizan con mayor frecuencia, para nuestro ejemplo dentro de los parámetros usados para consultas encontramos ya algunos índices implícitos como “id” que permiten un acceso más rápido, estos corresponden a índices creados de manera automática por Oracle.
* CLIENTES tiene una clave primaria en la columna “id\_clientes”.



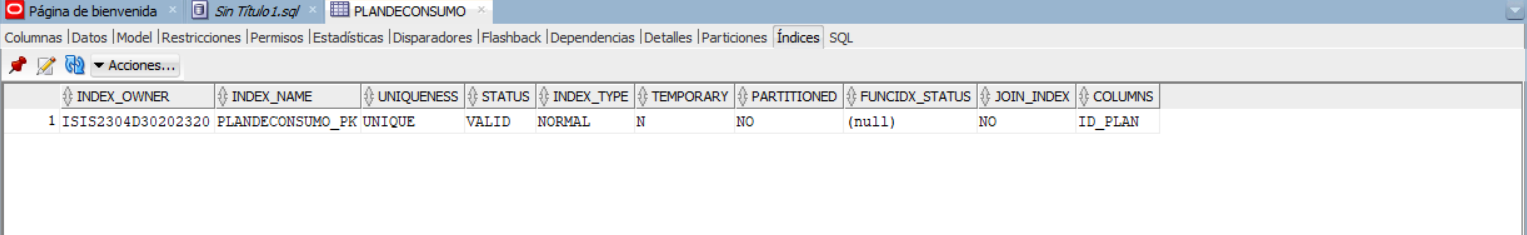
* HABITACION tiene una clave primaria en la columna “id\_habitacion”.



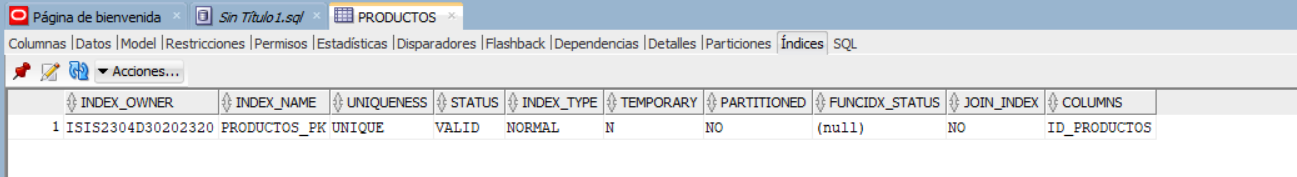
* HOTEL tiene una clave primaria en la columna “id\_hotel”.



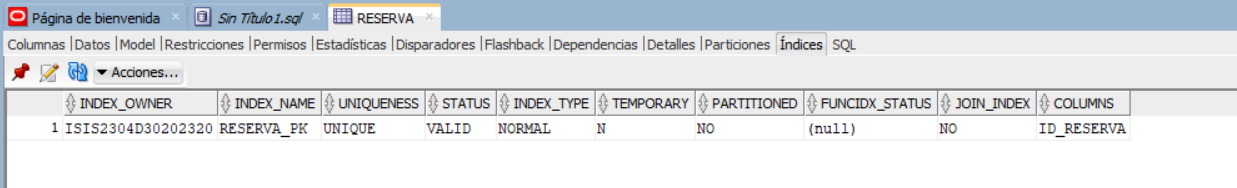
* PLANDECONSUMO tiene una clave primaria en la columna “id\_plan”.



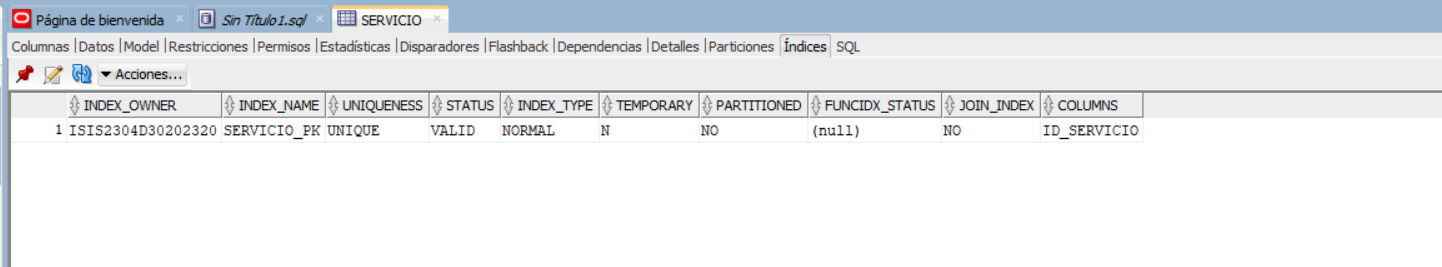
* PRODUCTOS tiene una clave primaria en la columna “id\_productos”.



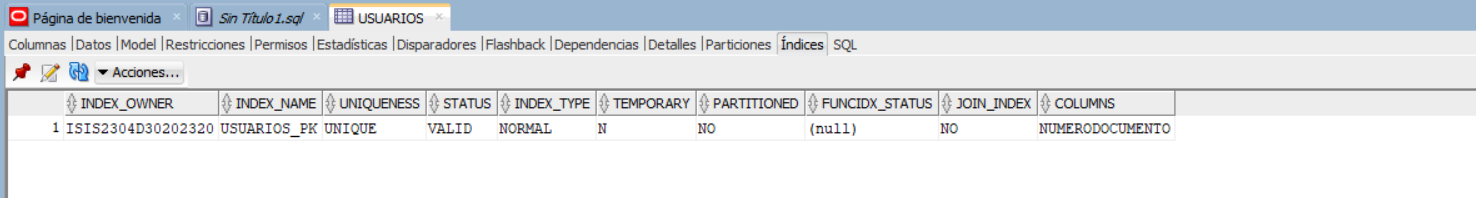
* RESERVA tiene una clave primaria en la columna “id\_reserva”.



* SERVICIO tiene una clave primaria en la columna “id\_servicio”.



* USUARIOS tiene una clave primaria en la columna “numerodocumento”.



**RF7 y RF8: implementar las operaciones de reserva de alojamiento o servicios por parte de un cliente: para cada operación se supone que se cumplen todas las condiciones para poderla realizar.**

* Para este caso necesitamos que las consultas sobre la disponibilidad sean más eficientes, por ende, se necesitarían poner índices en columnas con relación al tipo de servicio y la fecha.
* Para el entorno de Oracle el indicie más común y que garantiza optimización en tiempos de búsqueda es el B+Tree, sumado a un tipo de índice secundario que acelera las búsquedas en las columnas como el tipo de habitación, la capacidad, el nombre del servicio y el horario.
* Con estos índices buscaríamos cumplir que las columnas de “fechaEntrada” y “fechaSalida” cuenten ya con un alto nivel de selectividad, usando para esto el caso de que cada fecha o es única, o cuenta con un limite de repeticiones. Deberíamos como ejemplo una temporada de vacaciones de 90 días con todas las fechas registradas.
* Selectividad aproximada = 1 / 90 = 0,01
* 1.11% Es decir seria útil utilizar índices ya que la selectividad aproximada es menos al 25%.

**Algunos ejemplos de índices:**

**CREATE INDEX idx\_servicio\_horario ON servicio (horario);**

/ Para crear un indice en la tabla SERVICIO con columna “horrario”, esto con el fin de consultar por horarios los distintos servicios disponibles.

**CREATE INDEX idx\_habitacion\_capacidad ON habitacion (capacidad);**

/ Para crear un indice en la tabla HABITACION con columna “capacidad”, con el fin de consultar la capacidad de una habitación.

**• RF9 y RF10: Implementar las operaciones de llegada de un cliente al hotel y de sus consumos, suponiendo que todas las condiciones de validez se cumplen.**

* Estas operaciones de llegada y consumo representan información valiosa y frecuente dentro de las consultas. La creación de índices en las columnas relevantes a HABITACION y SERVICIO mejoraran el rendimiento de las consultas.
* La forma en la que están definidos estos índices es en B+tree, sumado a un índice primario ya que utilizar llaves con ese sentido, un índice simple ya que no necesita de varios datos y por último disperso.
* La creación de un índice se justifica tanto en el uso de fechas, ya que lo hace con una selectividad optima y por otra parte también funciona para la cantidad de servicios disponibles. Con esto último decimos que: Del total de servicios ofrecidos que son 12 y siendo un uso de al menos un servicio por día: 1080 veces que se usó el servicio en una temporada vacacional, tendríamos entonces, que la selectividad aproximada seria de 1 / 1080 = 0.000925...
* Que equivale al 0.09 %, con lo que tendríamos que ser un índice útil según el criterio de selectividad.
* **Algunos ejemplos:**

**CREATE INDEX idx\_habitacion\_fecha\_disponibilidad ON habitacion(fecha\_disponibilidad);**

**CREATE INDEX idx\_habitacion\_capacidad ON habitacion(capacidad);**

**CREATE INDEX idx\_habitacion\_estado ON habitacion(estado);**

// Estos indicen aceleran la busqueda de habitaciones disponibles en una fecha especificada.

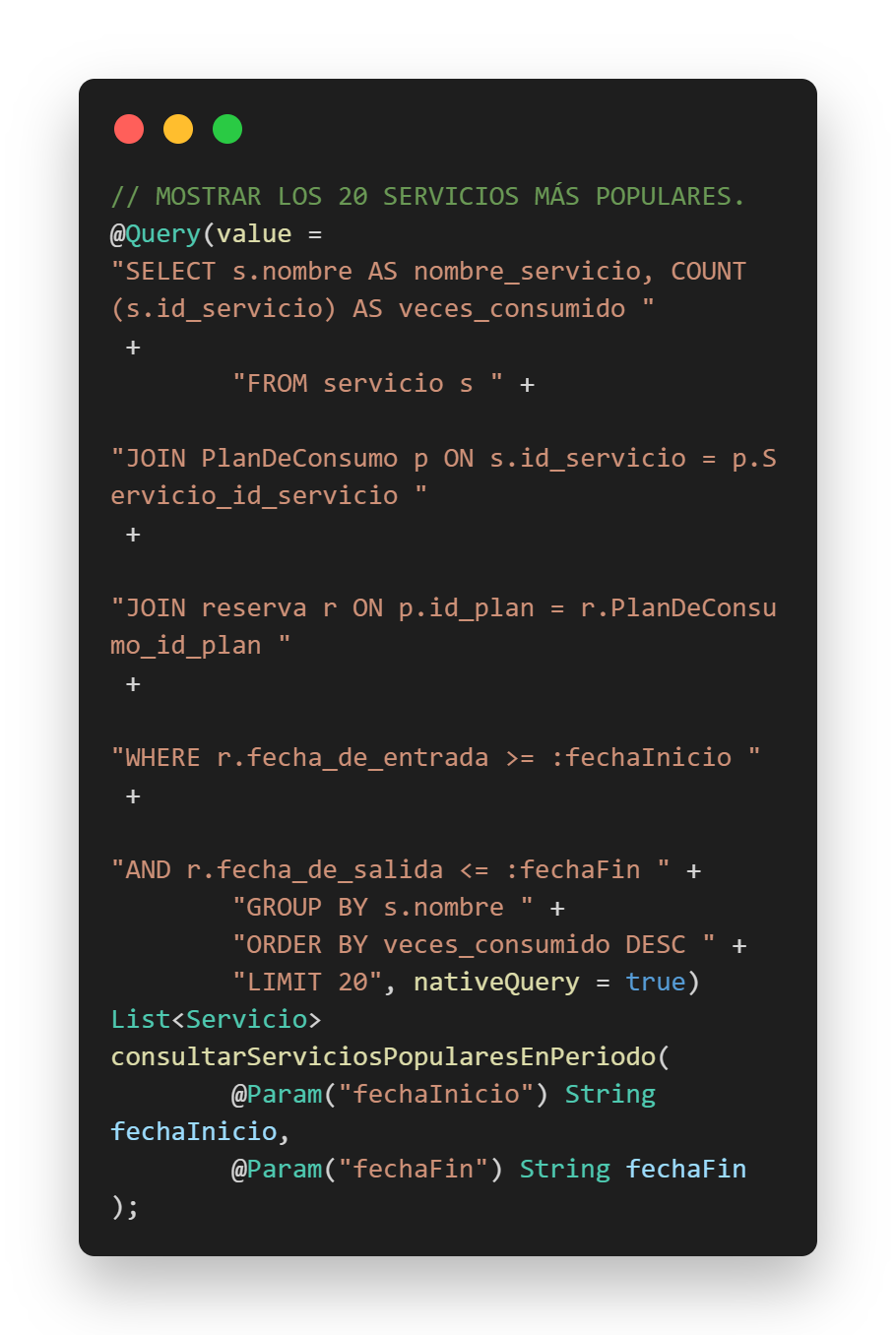
**RF11: Implementar la operación de salida de un cliente del hotel, suponiendo que todas las condiciones de validez se cumplen.**

* Las operaciones de salida se encuentran más ligadas a la actualización de los datos, más que a una consulta como en los requerimientos previos. Si suponemos que todas las condiciones de validez se cumplen, no tendríamos un estimado de selectividad aproximado ya que no contamos con datos distintos que evaluar y a su vez no contaríamos con más campos que los creados por las tablas, lo que nos deja con un estimado mayor al 25% requerido.

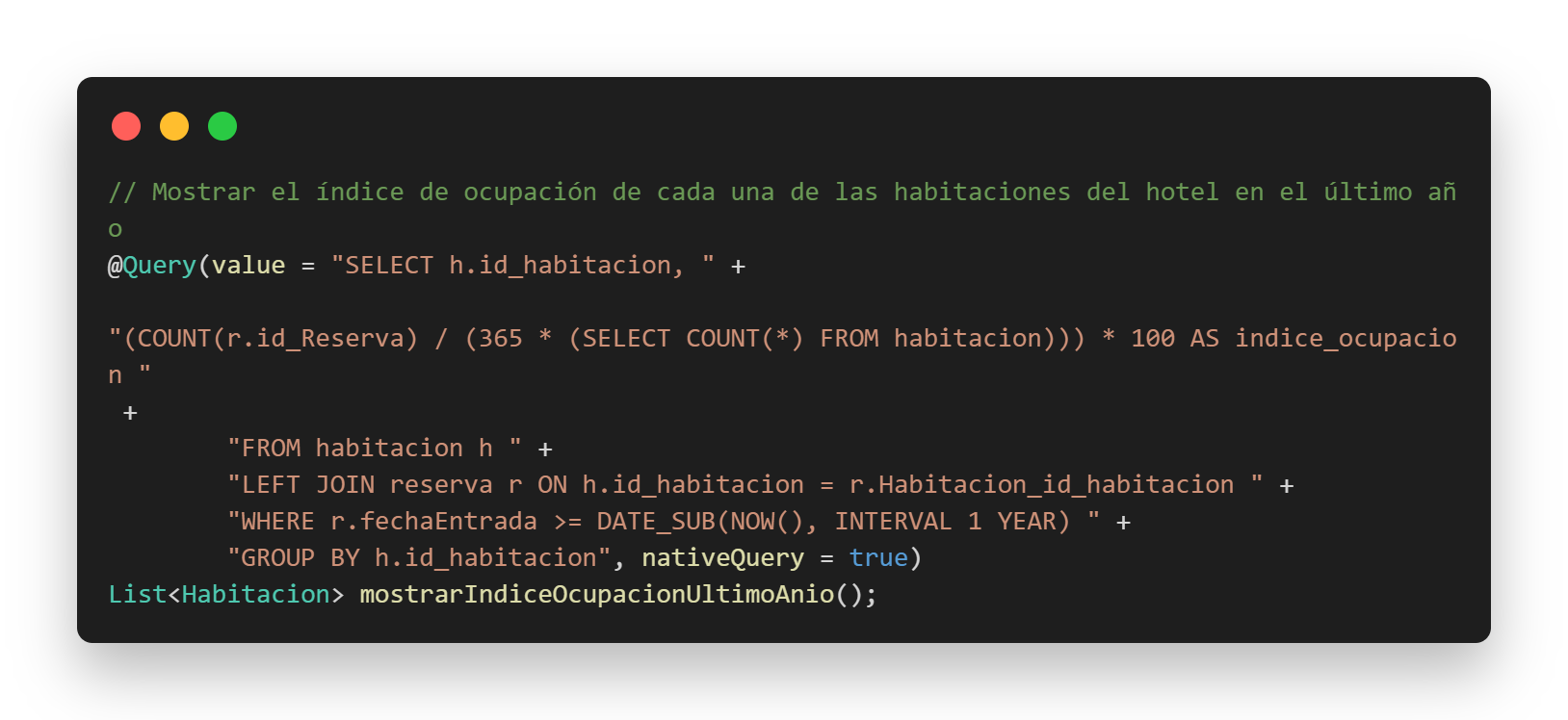
### (18%) Diseño de las consultas: para cada requerimiento funcional de consulta solicitado, documente:

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

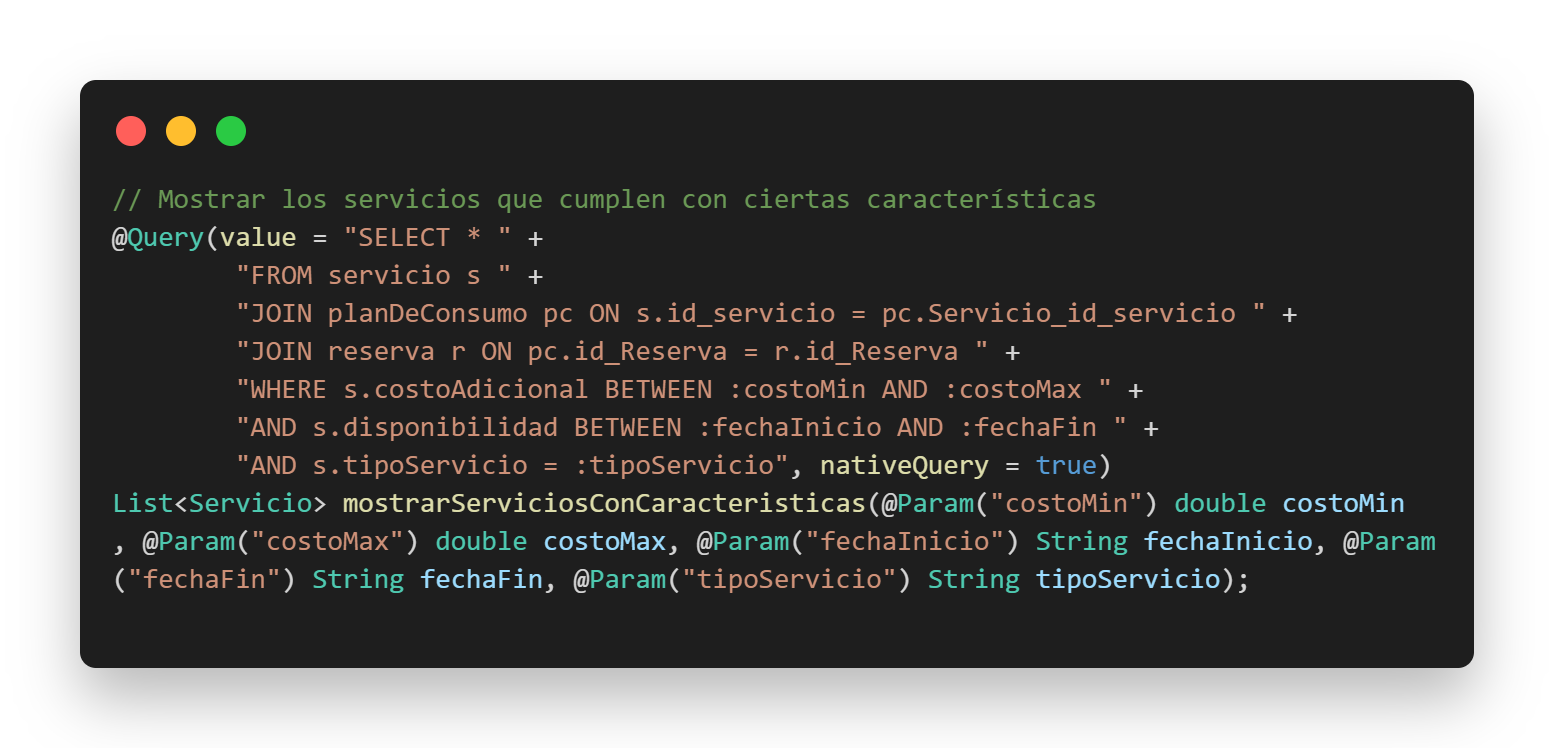


RFC2 - MOSTRAR LOS 20 SERVICIOS MÁS POPULARES.

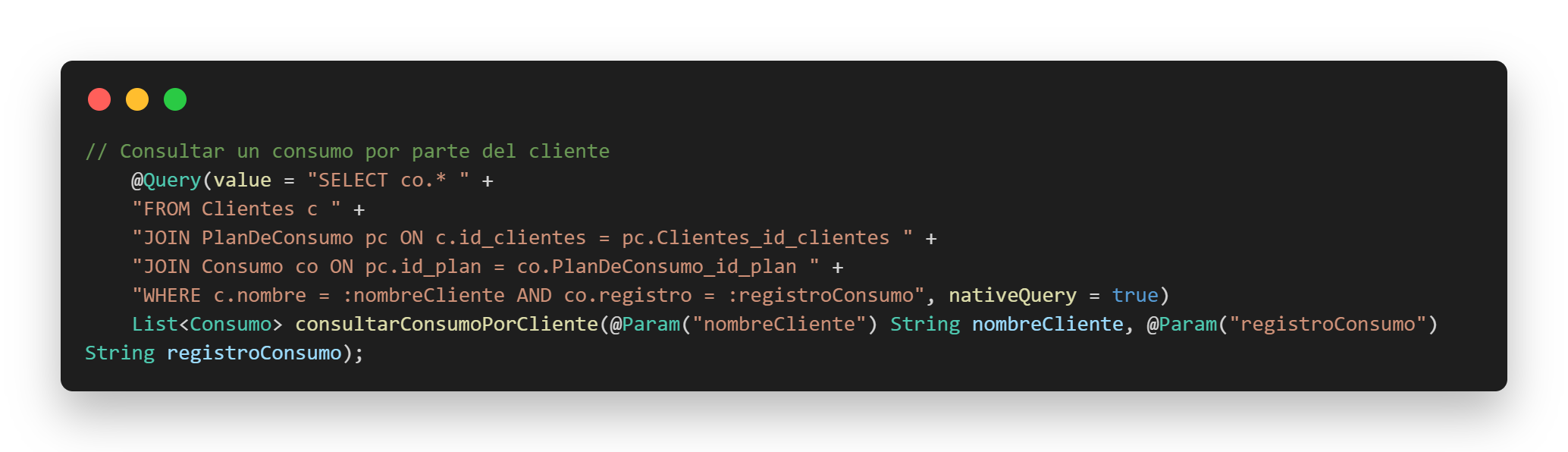
RFC3 - MOSTRAR EL ÍNDICE DE OCUPACIÓN DE CADA UNA DE LAS HABITACIONES DEL HOTEL



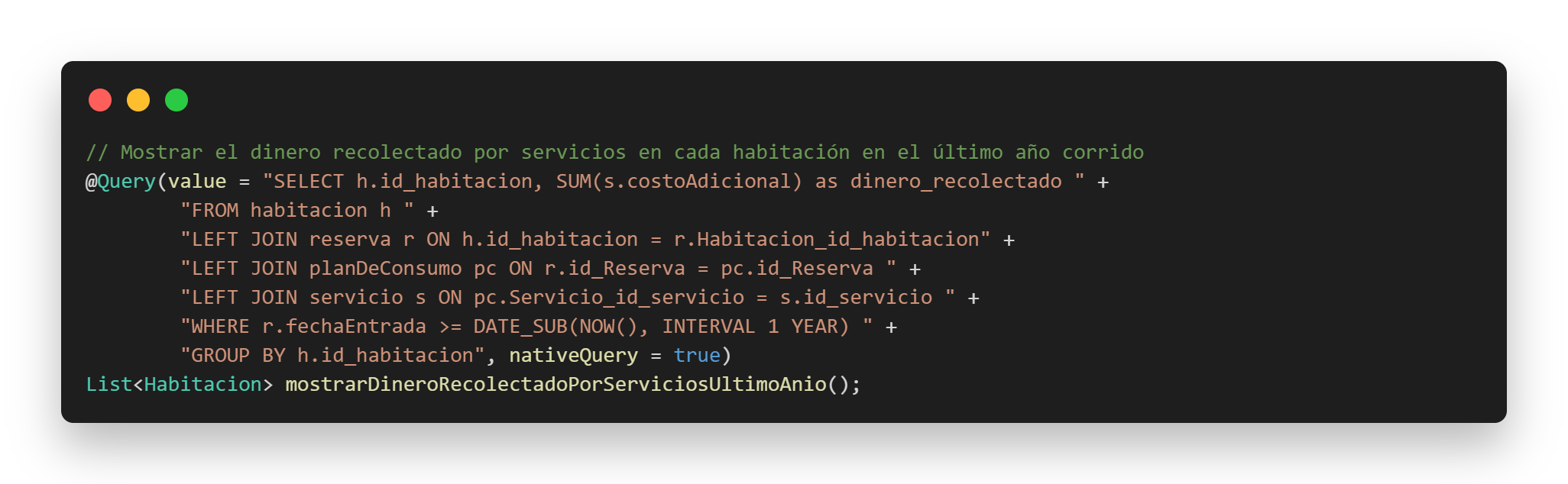
RFC4 - MOSTRAR LOS SERVICIOS QUE CUMPLEN CON CIERTA CARACTERÍSTICA

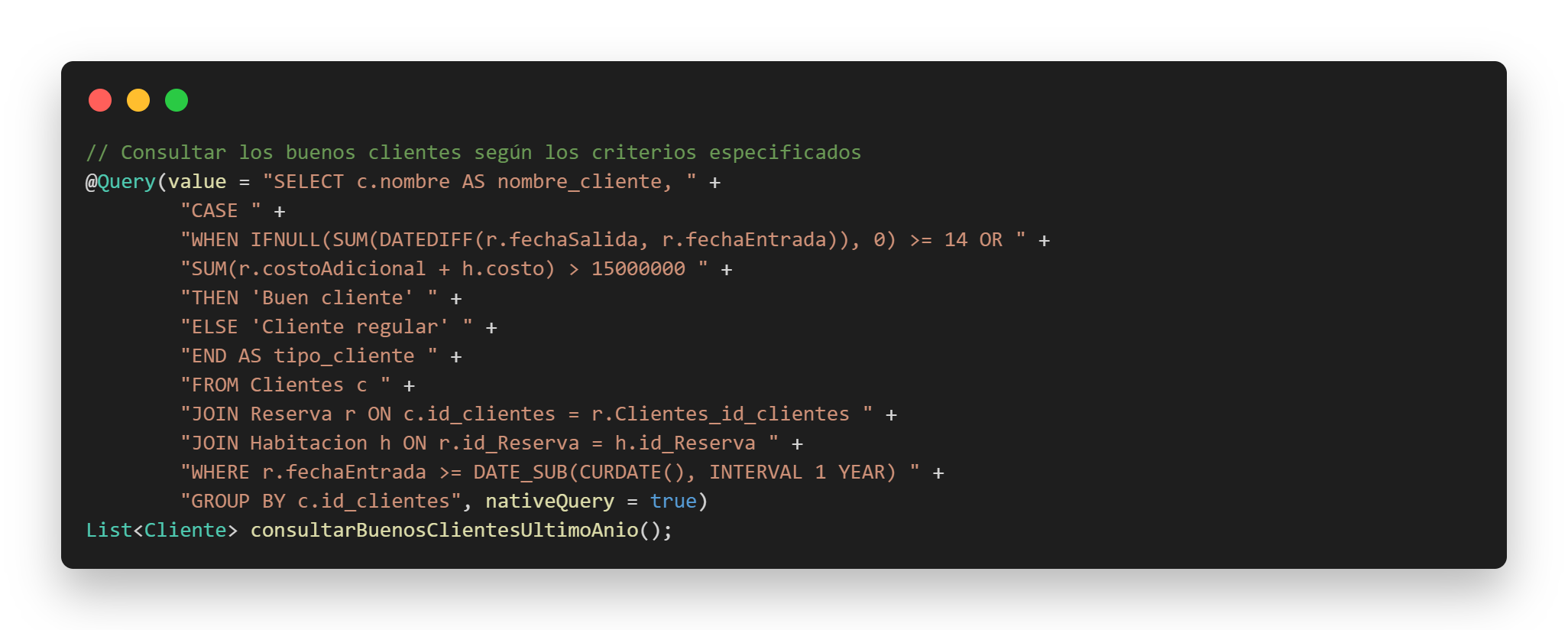


RFC5 - MOSTRAR EL CONSUMO EN HOTELANDES POR UN USUARIO DADO, EN UN RANGO DE FECHAS INDICADO.

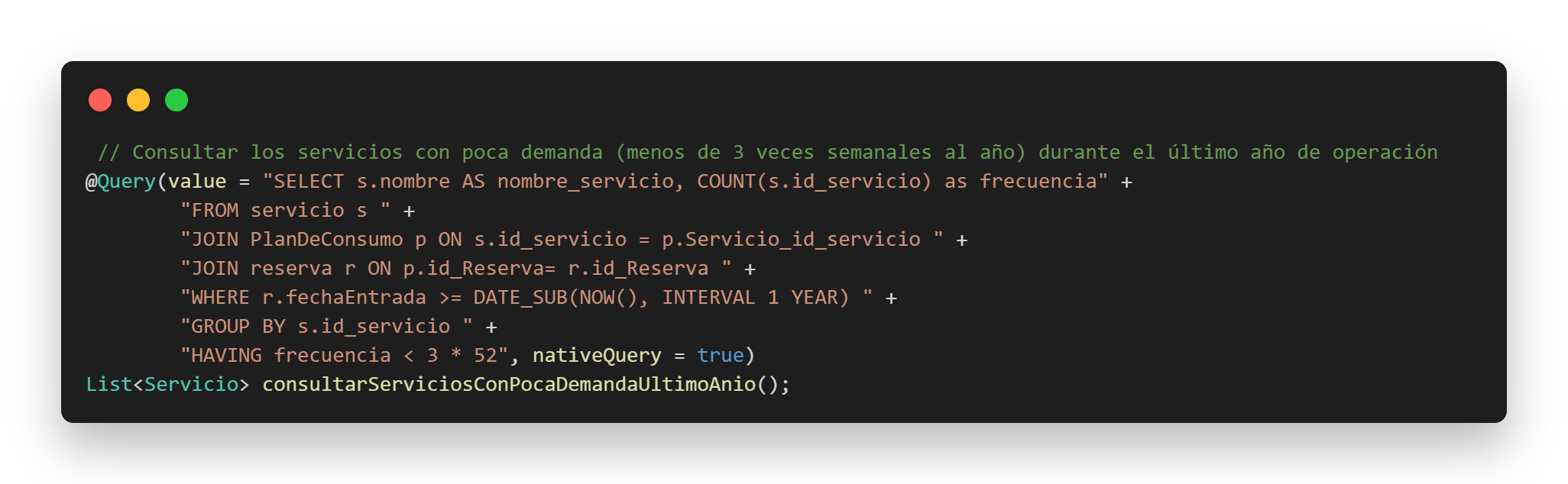


RFC6 - ANALIZAR LA OPERACIÓN DE HOTELANDES

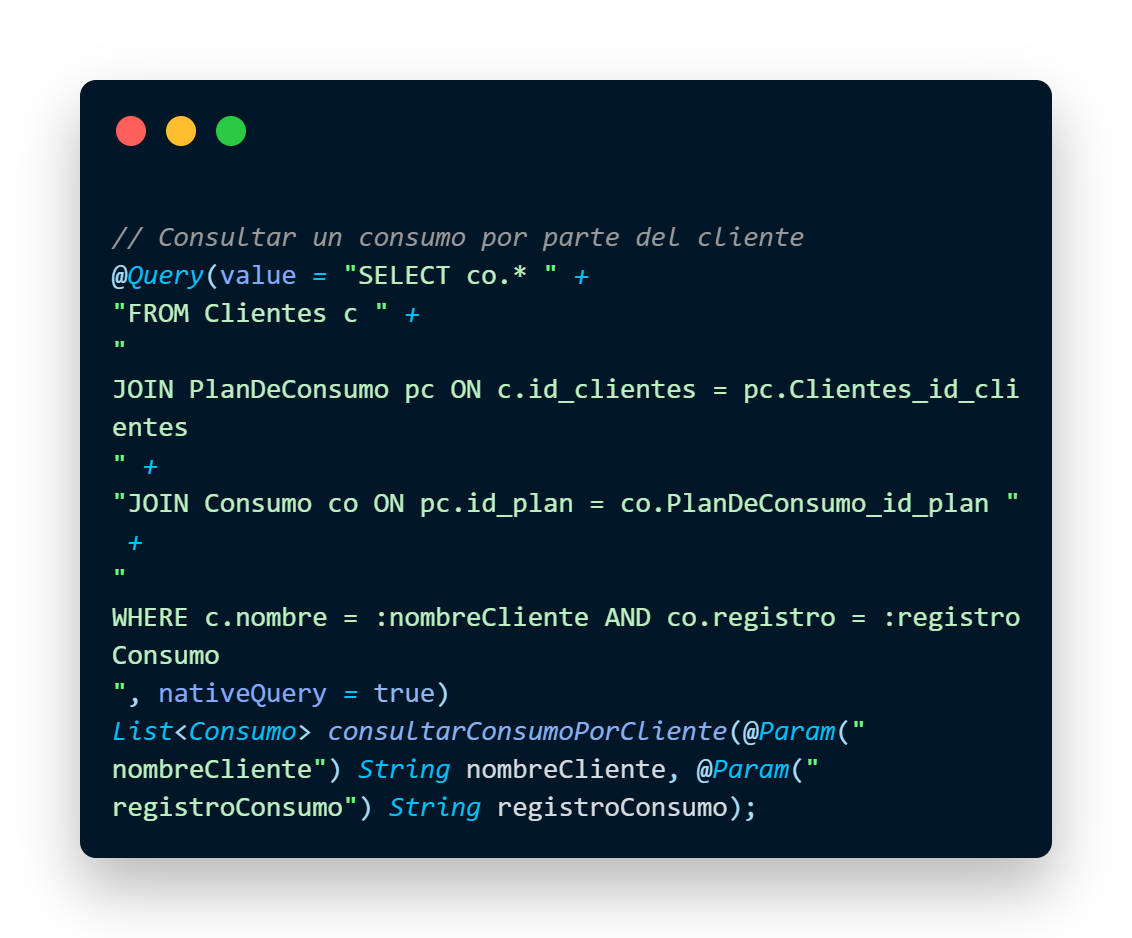


RFC7 - ENCONTRAR LOS BUENOS CLIENTES

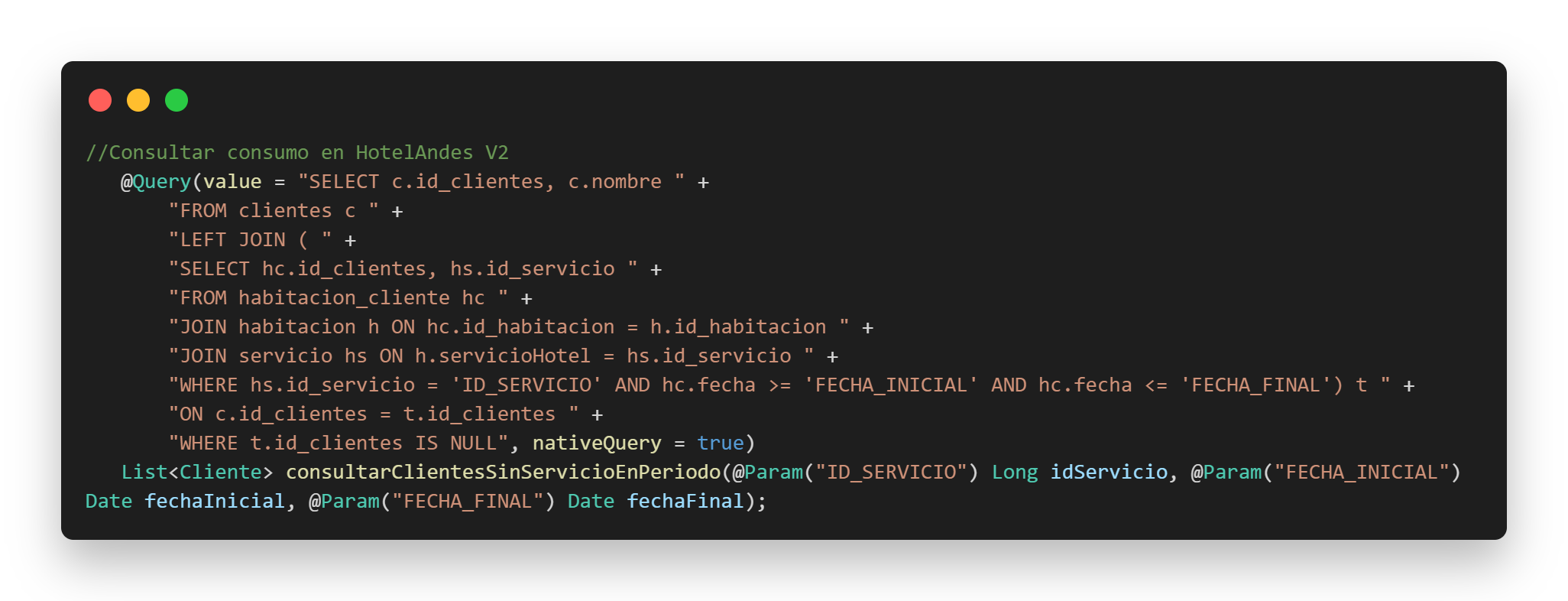
RFC8 - ENCONTRAR LOS SERVICIOS QUE NO TIENEN MUCHA DEMANDA



RFC9 - CONSULTAR CONSUMO EN HOTELANDES



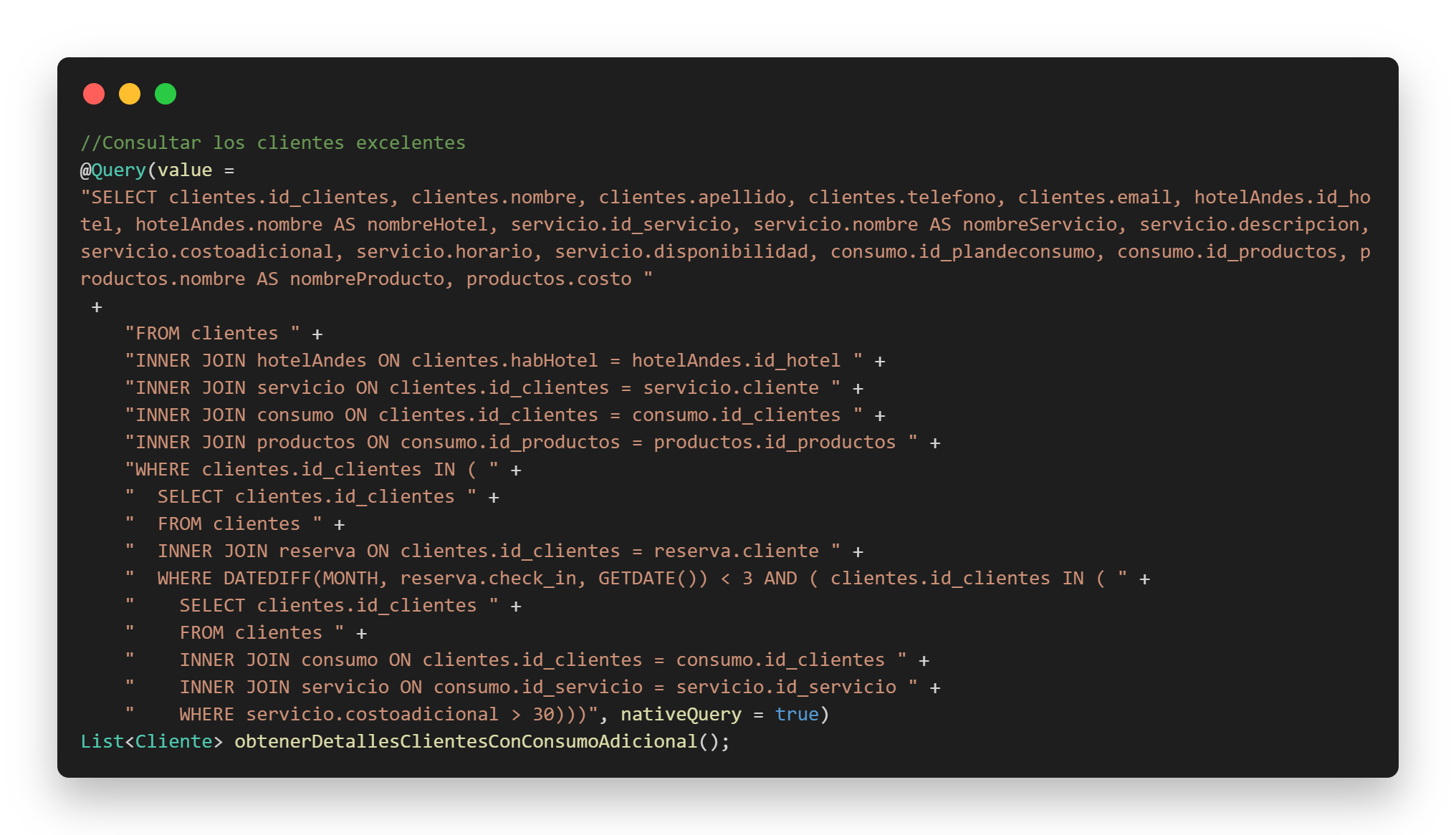
RFC10 - CONSULTAR CONSUMO EN HOTELANDES – RFC9-V2



RFC11 - CONSULTAR FUNCIONAMIENTO



RFC12 - CONSULTAR LOS CLIENTES EXCELENTES



## (25%) Diseño y cargue masivo de datos

Para la carga de datos se realizó un script con el elnguaje de programación Java, para que este fuese agregado directamente al repositorio. El programa utiliza la biblioteca JDBC de Oracle para interactuar con la base de datos, solicitando las credenciales y estableciendo la conexión. Luego se realiza las inserciones de los datos que fueron generados. Por último, el programa cierra la conexión y imprime en terminal:

// System.out.println(("Datos de prueba insertados en la tabla clientes.");

