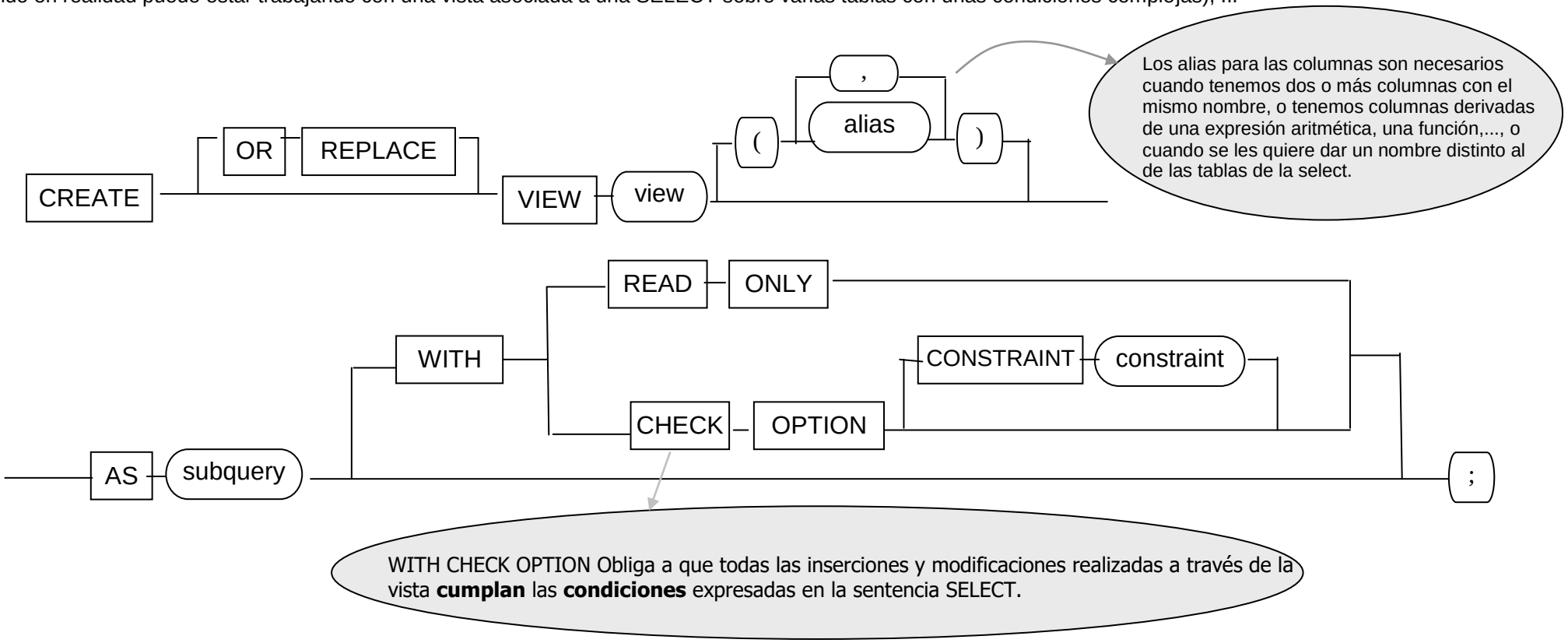
# Sesión 4: Introducción a vistas, índices, y permisos

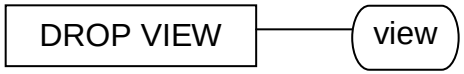
Una vista es una tabla "lógica" asociada a una sentencia SELECT, es decir, físicamente no contiene sus propios datos ni estructura, sino que cada vez que hacemos referencia a la vista, la sentencia SELECT asociada a la misma se ejecuta. Si ejecutamos SELECT * FROM nombre_vista en dos instantes distintos, el resultado que obtengamos puede variar, ya que puede haber variado el contenido de las tablas asociadas a la a la vista y por lo tanto el resultado de la SELECT.

Las vistas son muy útiles por motivos de seguridad, o por simplificar la estructura de tablas extensas a un determinado usuario (te permite trabajar con una "tabla" cuando en realidad puede estar trabajando con una vista asociada a una SELECT sobre varias tablas con unas condiciones complejas), ...




Sobre una vista se pueden realizar operaciones como SELECT, INSERT, UPDATE y DELETE. El uso de estas operaciones está sujeto a restricciones derivadas de la forma en la que están creadas las vistas. Por ejemplo, no se puede utilizar INSERT sobre una vista creada con una función agregada (count(*), sum(nom_colum),...) o, por ejemplo, si falta en la vista la clave primaria de la tabla,…

Para borrar vitas




**Ejemplo:**

CREATE VIEW **tratamientocaro** AS

SELECT * FROM tratamiento WHERE precio> 60

Al no haber creado la vista con la restricción WITH CHECK OPTION, podríamos realizar la siguiente inserción:

INSERT INTO **tratamientocaro** *(Nota: TRATAMIENTO (código, observaciones, precio))*

VALUES ('EXMAS', 'Exfoliación y masaje completo', 50)

Si no se viola ninguna restricción (clave primaria, valor no nulo, check, …), la inserción se realiza sobre la **tabla base** de la vista (tratamiento). Al no haber puesto la restricción WITH CHECK OPTION no se valoran las condiciones de la sentencia SELECT asociada, sólo se mira la tabla asociada a la vista y sus restricciones.

Si a continuación hacemos

SELECT * from tratamiento la nueva fila se mostrará en el resultado porque se ha insertado en la tabla tratamiento.

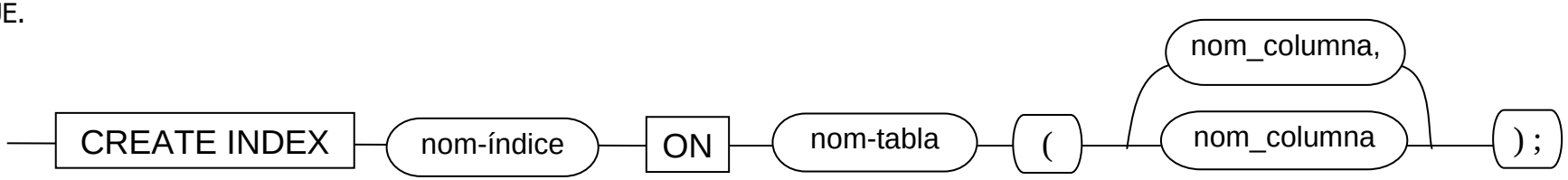
Si hacemos

SELECT * from tratamientocaro la nueva fila no aparecerá en el resultado ya que lo que se ejecuta en ese momento es la SELECT asociada a la creación de la vista. De hecho, por el momento, no hay ninguna fila que cumpla la condición.

*Un buen ejercicio a realizar podría ser valorar lo que ocurre con las inserciones cuando, en la sentencia SELECT asociada a la vista, la cláusula FROM nombra más de una tabla o cuando en la cláusula SELECT de la vista aparecen funciones agregadas (COUNT, SUM, …).*

## CREATE INDEX para definir un índice en una tabla

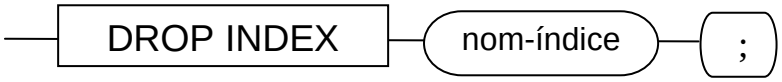
Un índice es una estructura de memoria secundaria que permite el acceso directo a las filas de una tabla. Hace que el tiempo de respuesta de una consulta disminuya, mejorando su rendimiento y optimizando su resultado. Su manejo se hace de forma inteligente, ya que es el propio Oracle quien decide qué índice se necesita. Por defecto Oracle siempre crea un índice para cada primaria que se defina y para cada restricción UNIQUE.




Ejemplo:

**CREATE INDEX ind_prov_cliente ON cliente(provincia);**

Para borrar un índice




*Cuando se ejecuta una sentencia en Oracle, existe un módulo "optimizador" que analiza varias estrategias y elige una de ellas, no necesariamente la mejor, sino una lo suficientemente buena según un cierto umbral. Por lo tanto, no siempre que tengamos definido un índice se va a utilizar, aunque a lo mejor utilizándolo la estrategia obtenida sería la mejor posible a la hora de ejecutar una determinada acción. Por otro lado, siempre que se haya definido un índice significa que las operaciones de inserción, modificación o borrado en una tabla llevarán un coste adicional de mantenimiento del índice. Por todo lo explicado no se definen índices para todas las columnas de una tabla.*

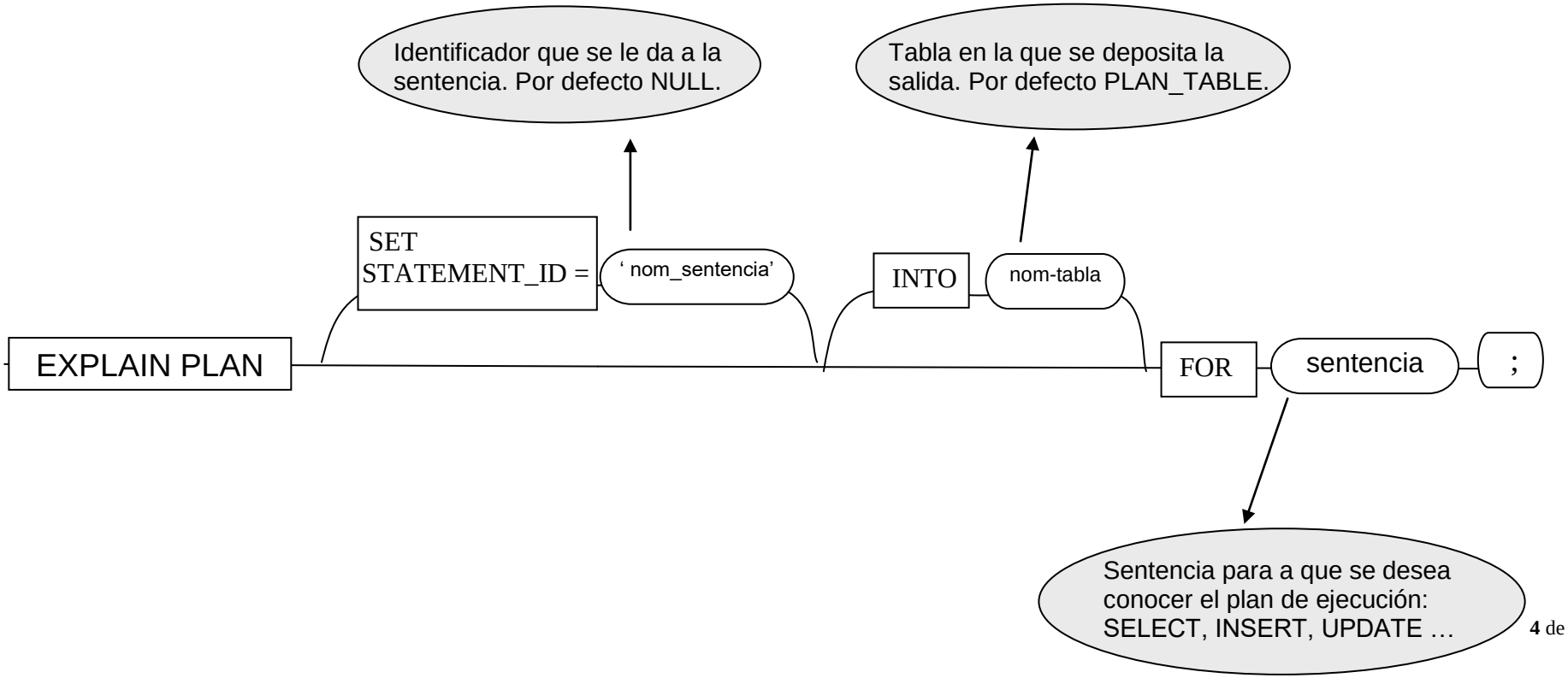
*¿Cómo podemos saber si en una consulta se está usando (o no) un índice que hemos definido? … Utilizando la sentencia* ***EXPLAIN PLAN***

## EXPLAIN PLAN para almacenar el plan de ejecución de una sentencia

Inserta una fila describiendo cada paso del plan de ejecución de una sentencia en la tabla PLAN_TABLE. La sentencia no se llega a ejecutar.

La tabla PLAN_TABLE tiene esta estructura:

- **Statement_id varchar(30): Valor opcional para identificar planes de ejecución**
- **Operation varchar(30): Nombre de la operación interna realizada**
- **Options varchar(225): Detalles sobre la operación**
- **Object_name varchar(30): Nombre de la tabla o índice**
- **Position numeric:**
  - **Para la primera fila indica el coste estimado por el optimizador para realizar la sentencia.**
  - **Para el resto, indica la posición relativa respecto al padre**



## Ejemplo:

***CASO 1: En la tabla CLIENTE sólo hay un índice definido***, el que corresponde a la clave primaria de la tabla. Ejecutemos las siguientes sentencias:

```
 explain plan
 set statement_id='cli1'
for select * from cliente where provincia='Alicante';

select operation, options, object_name, position
from plan_table
where statement_id='cli1'
```

```
OPERATION      OPTIONS        OBJECT_NAME          POSITION
---------------- ------------ ---------------------- ----------
SELECT STATEMENT                                        3
TABLE ACCESS     FULL         CLIENTE                   1
2 rows selected
```

No se utilizan índices

***CASO 2: Si definimos un índice sobre la columna provincia***

```
create index ind_prov_cliente  on cliente(provincia);
```

Y ahora ejecutamos:

```
explain plan
set statement_id='cli2'
for select * from cliente where provincia='Alicante';

select operation, options, object_name, position
from plan_table
where statement_id='cli2'
```

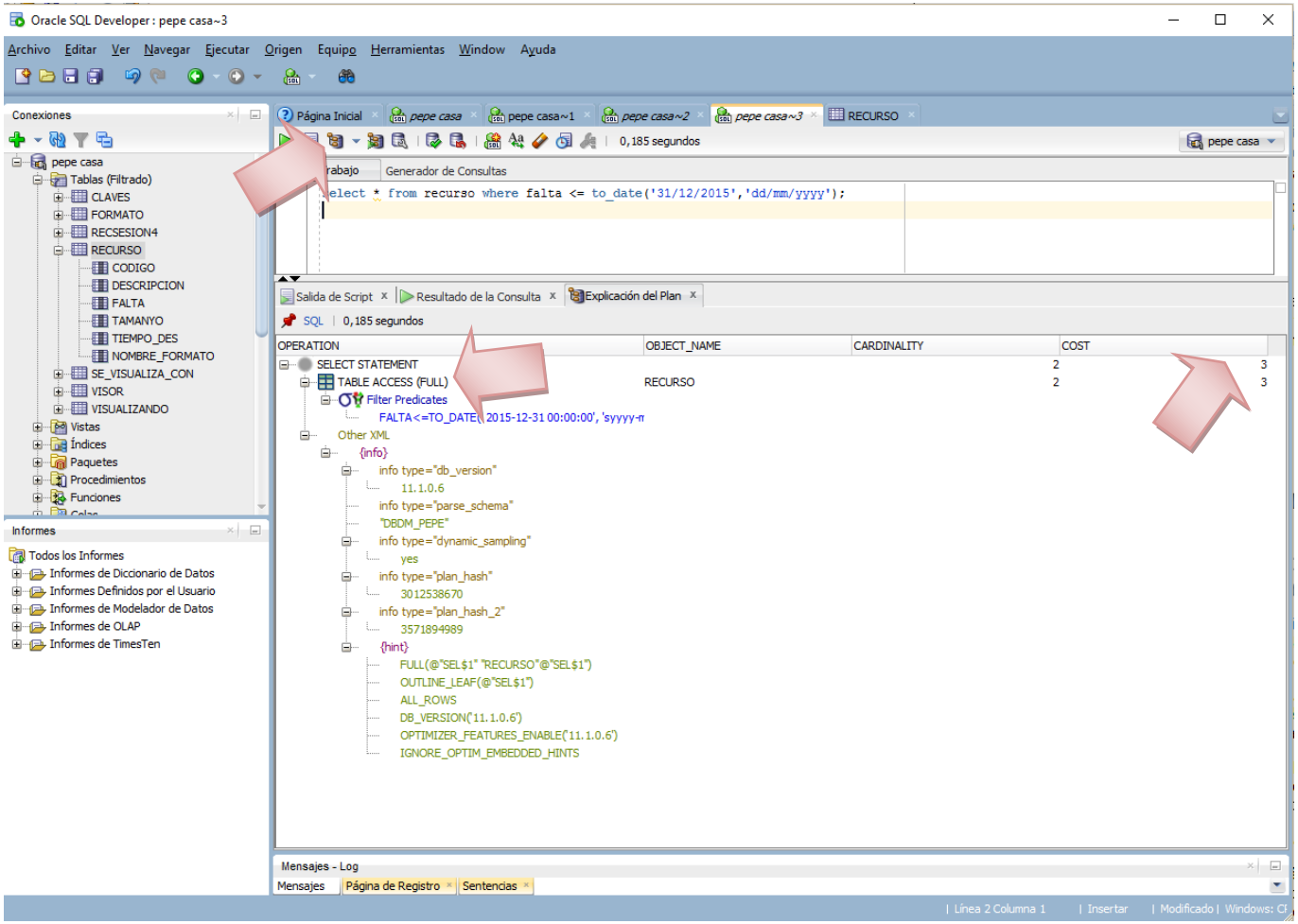
```
OPERATION      OPTIONS          OBJECT_NAME      POSITION
---------------- -------------- ------------------ ----------
SELECT STATEMENT                                      2
TABLE ACCESS     BY INDEX ROWID  CLIENTE             1
INDEX            RANGE SCAN      IND_PROV_CLIENTE    1
3 rows selected
```

Se utiliza el índice IND_PROV_CLIENTE

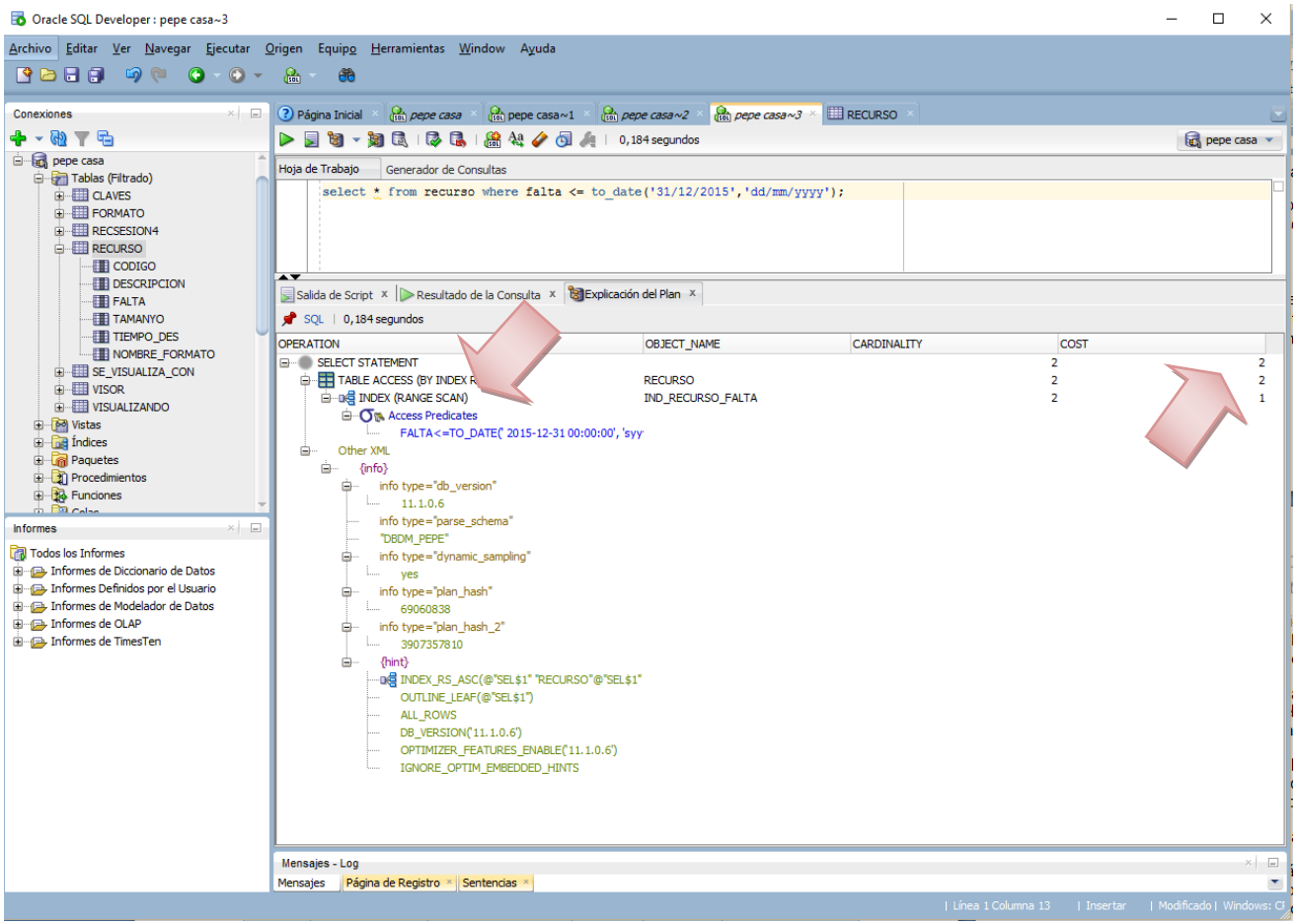
# Ejemplo usando SQL Developer

En las últimas versiones de SQL Developer podemos conocer el plan de ejecución de manera gráfica.

Sin usar un índice:

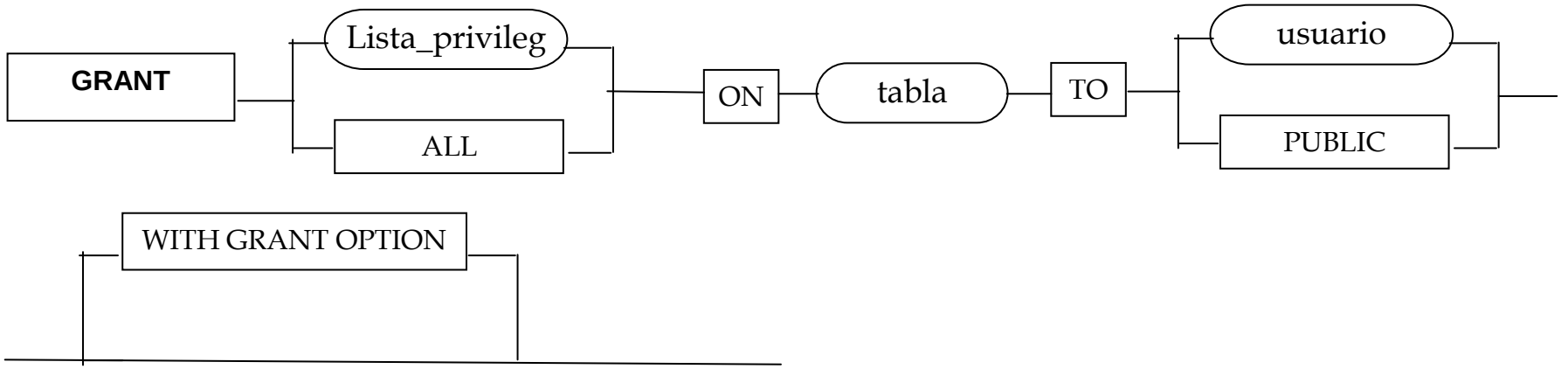


## Usando un índice:



## GRANT

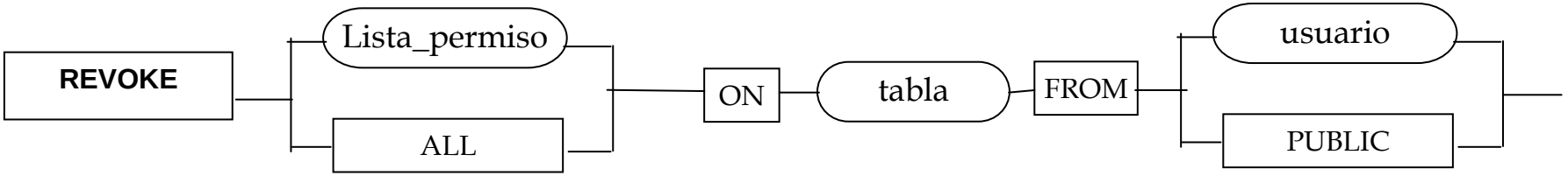
Sirve para otorgar privilegios. Se pueden definir privilegios sobre columnas concretas, y otra serie de opciones que aquí no están contempladas.




Lista de privilegios se refiere a SELECT, INSERT, UPDATE, ALTER, ...
Si ponemos la opción WITH GRANT OPTION, el usuario que recibe los privilegios los puede otorgar a su vez a otros usuarios.
Si en lugar de pober un nombre de usuario ponemos PUBLIC, se le otorgan a todos los usuarios.

## REVOKE

Sirve para retirar los permisos otorgados. Al igual que ocurre con GRANT, tiene muchas opciones que no están contempladas en este pequeño resumen.



# USOS PRÁCTICOS

**VISTAS:**

Algunos pueden ser:

a) Simplificación de tablas: acotando las columnas y filas de una o varias tablas mediante una vista, simplificaremos el acceso a esos datos por parte de algunos usuarios.
b) Reutilización de consultas: si hay consultas complejas que realizamos frecuentemente, creando una vista facilitaremos la ejecución de esa consulta. Obtendremos la información tan solo ejecutando una simple select hacia ella.
c) Seguridad de acceso: podremos ocultar datos a ciertos usuarios (tanto columnas como filas de una o varias tablas).

**ÍNDICES:**

Sirven para acelerar las consultas a las tablas. Teniendo una buena estructura de índices, podemos llegar a acelerar en gran medida las consultas a nuestro modelo de datos. Las herramientas de planificación de consultas que nos ofrecen los sistemas de base de datos nos dan pistas acerca de cómo poder acelerarlas mediante el uso de ínidices.

**GRANT y REVOKE:**

Permiten que los usuarios que nosotros queremos accedan de la forma que nosotros queremos a los datos que nosotros queremos. Proporcionan simplicidad y seguridad a la información.