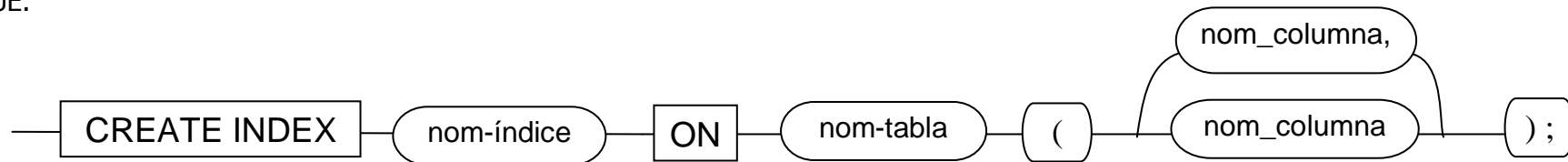




## Sesión 13

### CREATE INDEX para definir un índice en una tabla

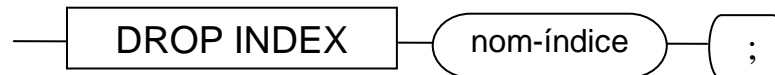
Un índice es una estructura de memoria secundaria que permite el acceso directo a las filas de una tabla. Hace que el tiempo de respuesta a una consulta disminuya, mejorando su rendimiento y optimizando su resultado. Su manejo se hace de forma inteligente. Es el propio Oracle quien decide qué índice 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, el módulo "optimizador" 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.*

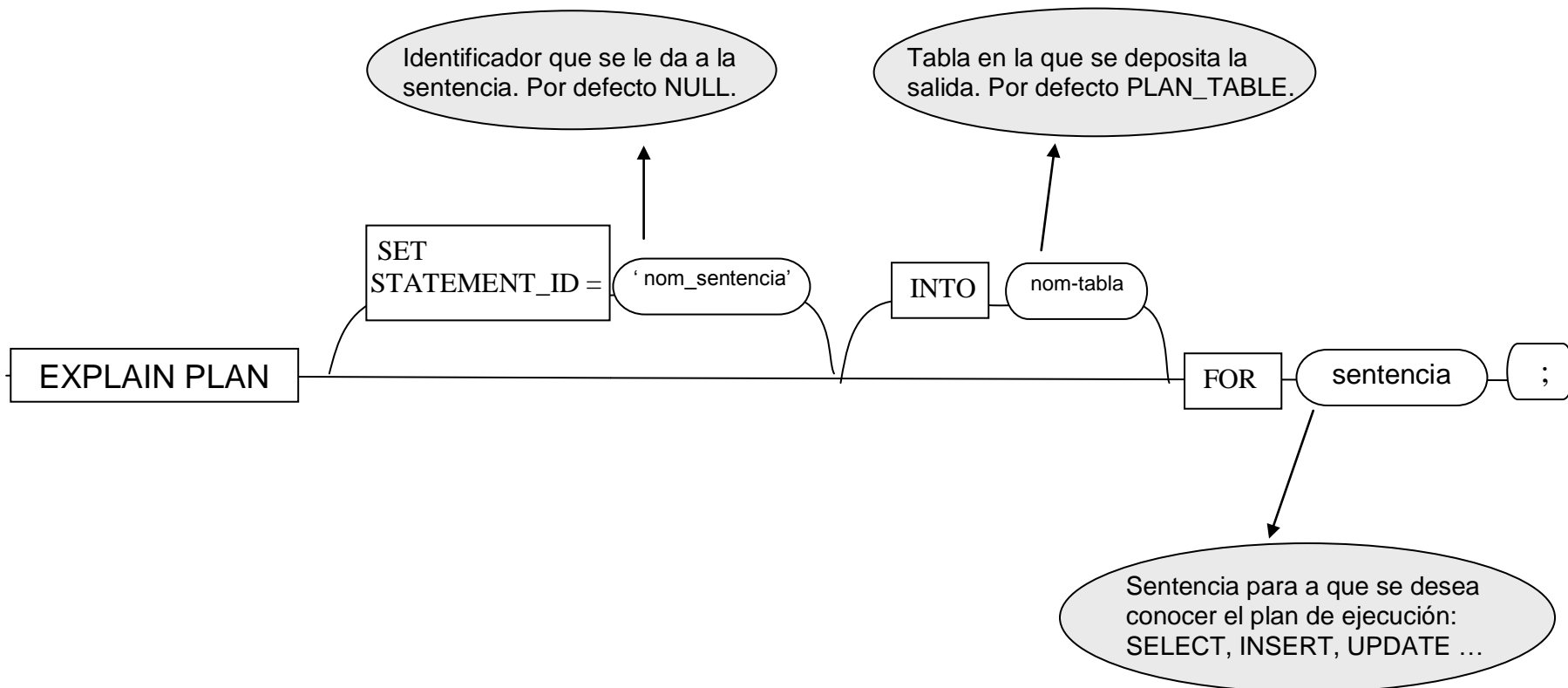
*¿Cómo podemos saber si en una consulta se está utilizando, o no, un índice que hemos definido? ... Apoyándonos en **EXPLAIN PLAN***

## 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 contiene

- **Statement\_id varchar2(30)** Valor opcional para identificar planes de ejecución
- **Operation varchar2(30)** Nombre de la operación interna realizada
- **Options varchar2(225)** Detalles sobre la operación
- **Object\_name varchar2(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. Ejecutamos 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

Las últimas versiones de SQLDeveloper nos permiten ver el plan de ejecución de manera gráfica

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

The screenshot shows the SQL Developer interface with the following components:

- Query Builder:** Contains the query `select * from cliente where provincia='Alicante'`.
- Execution Plan:** A table showing the execution plan for the query. The table has columns: OPERATION, OBJECT\_NAME, OPTIONS, CARDINALITY, and COST.
- Execution Plan Table:**

OPERATION	OBJECT_NAME	OPTIONS	CARDINALITY	COST
SELECT STATEMENT			3	2
TABLE ACCESS	CLIENTE	FULL	3	2

Below the execution plan table, there is a section for 'Other XML' and 'Hints'. The 'Hints' section shows the following hints:

- `FULL(@SEL$1 "CLIENTE"@"SEL$1")`
- `OUTLINE_LEAF(@"SEL$1")`
- `ALL_ROWS`
- `DB_VERSION("12.1.0.2")`
- `OPTIMIZER_FEATURES_ENABLE("12.1.0.2")`
- `IGNORE_OPTIM_EMBEDDED_HINTS`

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

The screenshot shows the SQL Developer interface. The top toolbar has the 'Run' button circled in red. The main window displays the following SQL query:

```
create index ind_prov_cliente on cliente(provincia);  
select * from cliente where provincia='Alicante';
```

Below the query, the 'Explain Plan' window is open, showing the execution plan for the SQL statement. The plan is as follows:

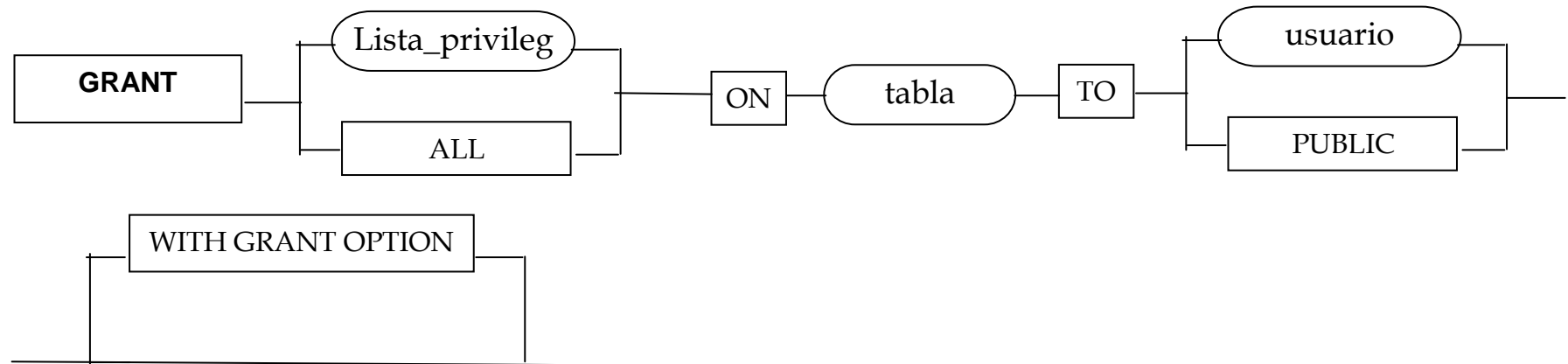
OPERATION	OBJECT_NAME	OPTIONS	CARDINALITY	COST
SELECT STATEMENT				2
TABLE ACCESS	CLIENTE	BY INDEX ROWID BATCHED		2
INDEX	IND_PROV_CLIENTE	RANGE SCAN		1

Red circles highlight the 'TABLE ACCESS' and 'INDEX' steps in the plan, the 'BY INDEX ROWID BATCHED' option, and the cost values (2, 2, 1). The 'Access Predicates' section shows 'PROVINCIA='Alicante''. The 'Other XML' section shows the following information:

```
{info}  
  info type="db_version"  
    12.1.0.2  
  info type="parse_schema"  
    "DBO_EVA16"  
  info type="plan_hash_full"  
    319299624  
  info type="plan_hash"  
    3638511148  
  info type="plan_hash_2"  
    319299624  
{hint}  
  BATCH_TABLE_ACCESS_BY_ROWID(@"SEL$1" "CLIENTE"@"SEL$1")  
  INDEX_RS_ASC(@"SEL$1" "CLIENTE"@"SEL$1" ("CLIENTE"."PROVINCIA"))  
  OUTLINE_LEAF(@"SEL$1")  
  ALL_ROWS  
  DB_VERSION("12.1.0.2")  
  OPTIMIZER_FEATURES_ENABLE("12.1.0.2")  
  IGNORE_OPTIM_EMBEDDED_HINTS
```

## GRANT

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



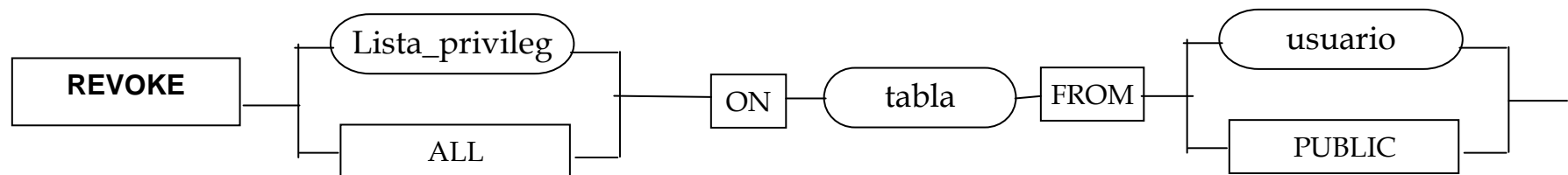
Lista de privilegios se refiere a SELECT, INSERT, UPDATE, ALTER, ...

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 poner un nombre de usuario ponemos PUBLIC, se le otorgan a todos los usuarios.

## REVOKE

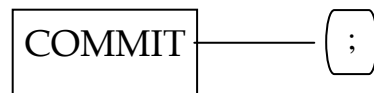
Sirve para retirar los privilegios otorgados. Al igual que ocurre con GRANT, tiene muchas opciones que no están contempladas en este pequeño resumen.



## **COMANDOS PARA EL CONTROL DE TRANSACCIONES. (Va asociado al concepto de transacción, en clase de teoría)**

### **COMMIT**

Cuando estamos ejecutando sentencias contra la base de datos, éstas no se hacen efectivas hasta que no ejecutamos un COMMIT. Hay algunas sentencias que llevan COMMIT asociado, por ejemplo CREATE TABLE. También implica un COMMIT abandonar una sesión de trabajo.



### **SAVEPOINT**

*Para marcar un punto dentro de un conjunto de sentencias que se están ejecutando.*



### **ROLLBACK**

Para volver atrás, deshacer acciones que se han realizado. Si ponemos savepoint se retrocede hasta ese punto. Si no especificamos savepoint se deshace toda la transacción.

