Investiga en w3schools el uso de EXISTS, NOT EXISTS, ANY y ALL.

https://www.w3schools.com/sql/sql_exists.asp

https://www.w3schools.com/sql/sql_any_all.asp

Exists y NOT Exists

Los operadores "exists" y "not exists" se emplean para determinar si hay o no datos en una lista de valores.

Estos operadores pueden emplearse con subconsultas correlacionadas para restringir el resultado de una consulta exterior a los registros que cumplen la subconsulta (consulta interior). Estos operadores retornan "true" (si las subconsultas retornan registros) o "false" (si las subconsultas no retornan registros).

Cuando se coloca en una subconsulta el operador "exists", Oracle analiza si hay datos que coinciden con la subconsulta, no se devuelve ningún registro, es como un test de existencia; Oracle termina la recuperación de registros cuando por lo menos un registro cumple la condición "where" de la subconsulta.

La sintaxis básica es la siguiente:

```
... where exists (SUBCONSULTA);
```

En este ejemplo se usa una subconsulta correlacionada con un operador "exists" en la cláusula "where" para devolver una lista de clientes que compraron el artículo "lapiz":

```
select cliente,numero
 from facturas f
 where exists
  (select *from Detalles d
    where f.numero=d.numerofactura
    and d.articulo='lapiz');
```

Puede obtener el mismo resultado empleando una combinación.

Podemos buscar los clientes que no han adquirido el artículo "lapiz" empleando "if not exists":

```
select cliente,numero
 from facturas f
 where not exists
  (select *from Detalles d
    where f.numero=d.numerofactura
    and d.articulo='lapiz');
```

Subconsultas any- some- all

"any" y "some" son sinónimos. Chequean si alguna fila de la lista resultado de una subconsulta se encuentra el valor especificado en la condición.

Compara un valor escalar con los valores de un campo y devuelven "true" si la comparación con cada valor de la lista de la subconsulta es verdadera, sino "false".

El tipo de datos que se comparan deben ser compatibles.

La sintaxis básica es:

```
...VALORESCALAR OPERADORDECOMPARACION
 any (SUBCONSULTA);
```

Queremos saber los títulos de los libros de "Borges" que pertenecen a editoriales que han publicado también libros de "Richard Bach", es decir, si los libros de "Borges" coinciden con ALGUNA de las editoriales que publicó libros de "Richard Bach":

```
select titulo
 from libros
 where autor='Borges' and
 codigoeditorial = any
  (select e.codigo
   from editoriales e
   join libros l
   on codigoeditorial=e.codigo
   where l.autor='Richard Bach');
```

La consulta interna (subconsulta) retorna una lista de valores de un solo campo (puede ejecutar la subconsulta como una consulta para probarla), luego, la consulta externa compara cada valor de "codigoeditorial" con cada valor de la lista devolviendo los títulos de "Borges" que coinciden.

"all" también compara un valor escalar con una serie de valores. Chequea si TODOS los valores de la lista de la consulta externa se encuentran en la lista de valores devuelta por la consulta interna.

Sintaxis:

```
  VALORESCALAR OPERADORDECOMPARACION all (SUBCONSULTA);
```

Queremos saber si TODAS las editoriales que publicaron libros de "Borges" coinciden con TODAS las editoriales que publicaron libros de "Richard Bach":

```
select titulo
 from libros
 where autor='Borges' and
 codigoeditorial = all
  (select e.codigo
   from editoriales e
   join libros l
   on codigoeditorial=e.codigo
   where l.autor='Richard Bach');
```

La consulta interna (subconsulta) retorna una lista de valores de un solo campo (puede ejecutar la subconsulta como una consulta para probarla), luego, la consulta externa compara cada valor de "codigoeditorial" con cada valor de la lista, si TODOS coinciden, devuelve los títulos.

Veamos otro ejemplo con un operador de comparación diferente:

Queremos saber si ALGÚN precio de los libros de "Borges" es mayor a ALGÚN precio de los libros de "Richard Bach":

```
select titulo,precio
 from libros
 where autor='Borges' and
 precio > any
  (select precio
   from libros
   where autor='Bach');
```

El precio de cada libro de "Borges" es comparado con cada valor de la lista de valores retornada por la subconsulta; si ALGUNO cumple la condición, es decir, es mayor a ALGÚN precio de "Richard Bach", se lista.

Veamos la diferencia si empleamos "all" en lugar de "any":

```
select titulo,precio
 from libros
 where autor='borges' and
 precio > all
  (select precio
   from libros
   where autor='bach');
```

El precio de cada libro de "Borges" es comparado con cada valor de la lista de valores retornada por la subconsulta; si cumple la condición, es decir, si es mayor a TODOS los precios de "Richard Bach" (o al mayor), se lista.

Emplear "= any" es lo mismo que emplear "in".

Emplear "<> all" es lo mismo que emplear "not in".

Resuelve propuestas en sql oracle del archivo any-some-all.sql y exists_not_exists.sql.