Como Construir un SELECT Eficiente en ORACLE en Solo 7 Pasos

¡Gratis!

Por Roberto Vicencio

Como Construir un SELECT Eficiente en ORACLE en Solo 7 Pasos

Índice del Contenido.

Introducción

Seguramente habrás pensado muchas veces que construir una consulta Select en Oracle es algo muy fácil y de hecho lo es si aprendes como se debe hacer correctamente. El problema ocurre cuando te enfrentas a un problema real que debes resolver y no sabes como plasmarlo dentro de un Select, ya que debes conseguir que una consulta te devuelva todos los datos que necesitas de una forma clara, limpia y ordenada. Además de todo esto, una consulta Select debe entregar los datos lo más rápidamente posible, de lo contrario, todo el trabajo que realizaste no habrá servido para nada. Lo anterior es posible si aprendes y aplicas metódicamente algunas técnicas básicas para construir Select que sean eficientes.

La Base de Datos Oracle se ha convertido en una herramienta indispensable para la mayoría de empresas a nivel mundial, lo que demuestra que Oracle es uno de los sistemas de base de datos relacional más completos, destacándose por su estabilidad, escalabilidad, soporte de transacciones y soporte multiplataforma. Por eso actualmente en el mercado existe una alta demanda de profesionales que sean expertos en esta materia y tú debes estar preparado para cubrirla satisfactoriamente y no quedarte afuera.

Como en todo aprendizaje se requiere mucha constancia y esfuerzo de tu parte, de modo que debes aceptar el desafío y comprometerte contigo mismo a alcanzar tu objetivo. Tan solo piensa en todo lo que podrías crecer profesionalmente en tu carrera, un ascenso, mayor renta o un mejor empleo… ¿suena realmente interesante, verdad?

En este breve documento te enseñare los 7 pasos fundamentales que tienes que saber para comenzar a construir consultas Select que sean eficientes, lo que consigo te llevará también a desarrollar software de mejor calidad.

Este va hacer el puntapié inicial para que finalmente te conviertas en un maestro de la Base de Datos Oracle, y aumentes tu productividad. Así que… no te demoro más con mis palabras, y ¡vamos a poner manos a la obra!

Paso N° 1

Aprende Como Utilizar el JOIN en tus Packages en Oracle

Saber utilizar correctamente la cláusula JOIN en las consultas SELECT de Oracle te permitirá desarrollar querys más eficientes, ya que podrás realizar solo una consulta a la Base de Datos Oracle en vez de varias consultas separadas lo que genera procesos que se demoran en entregar los resultados.

A muchos desarrolladores les es muy difícil aplicar el JOIN en sus consultas SELECT porque no entienden como funciona realmente, a mi personalmente me costo un poco aprender a dominarlo, pero una vez que lo entiendes te vuelves un adicto y lo utilizas siempre.

Veamos un ejemplo práctico para entender mejor como se debe utilizar el JOIN en las consultas SELECT:

Ejemplo #1: Supongamos que se necesita obtener datos de dos tablas para ser retornados por un procedimiento, veamos las dos formas de hacerlo:

```
Procedure Prc_Forma_Uno(pin_rut  in  number, pst_nom_persona  out  varchar2,
pst_nom_depto  out  varchar2) is

lin_id_departamento     number;

Begin

    select emp.nombre, emp.id_departamento into pst_nom_persona,

    lin_id_departamento

    from t_empleado emp

    where emp.rut = pin_rut;

    select dep.descripcion into pst_nom_depto

    from t_departamento dep

    where dep.id_departamento = lin_id_departamento;

 Exception

    When Others Then
```

```
        Pst_nom_persona:= 'Proceso terminado con error';

End Prc_Forma_Uno;
```

En esta primera forma se realizan 2 consultas a la Base de Datos, la primera para obtener el nombre de la persona y el id del departamento, luego se hace una segunda consulta para obtener el nombre del departamento.

```
Procedure Prc_Forma_Dos(pin_rut  in  number, pst_nom_persona  out  varchar2,
pst_nom_depto  out  varchar2) is

Begin

    select emp.nombre, dep.descripcion into pst_nom_persona, pst_nom_depto

    from t_departamento dep, t_empleado emp

    where emp.rut = pin_rut

        and dep.id_departamento = emp.id_departamento ;

 Exception

     When Others Then

         Pst_nom_persona:= 'Proceso terminado con error';

End Prc_Forma_Dos;
```

En esta segunda forma se realiza una sola consulta a Base de Datos porque se realiza un JOIN entre las tablas t_empleado y t_departamento por su campo en común que es el id_departamento (dep.id_departamento = emp.id_departamento).

Te propongo que pruebes de inmediato estos ejemplos de la cláusula JOIN en tu Base de Datos con tus propias tablas, procedimientos y funciones para que comiences a ejercitar la mano y aprendas a dominar esta técnica.

Paso N° 2

Cómo Utilizar la Cláusula GROUP BY en tus Packages en Oracle

Aprender a usar la cláusula GROUP BY de la sentencia SELECT de Oracle te permitirá construir querys agrupadas por uno o varios campos claves, realizando a la vez sumas de valores, contar la cantidad de registros, buscar el valor mayor y menor, sacar promedios, etc.

Generalmente la creación de grupos de datos se hace cuando se debe construir un reporte ya que siempre se pide mostrar un informe con la información resumida y otro con la información detallada. En los informes resumidos se muestran totales por día, por mes, por año, por cliente, por proveedor, por departamento, etc. Se pueden construir agrupaciones por una serie de criterios, todo depende del tipo de área o rubro de la compañía para la cual se construya el sistema.

Para usar correctamente la cláusula GROUP BY se debe conocer con anterioridad el modelo de los datos que se quiere agrupar, ya que primero debes determinar cuáles son los campos clave que se repiten en las tablas por los cuales se debe agrupar. Veamos un ejemplo práctico que permita entender mejor como usar la cláusula GROUP BY.

Ejemplo #1: Supone que debes hacer un reporte que entregue el total de tareas realizadas por cada empleado, de cada departamento, dentro del mes y por los 3 primeros meses del año. Veamos cómo se hace esta query:

```
/* Packages en Oracle */

Select

    Dep.nombre_dep departamento,

    Emp.nombre_emp empleado,

    To_char(tar.fecha_tarea,'mm-yyyy') Mes-Año,

    Count(tar.id_tareas_empleado) tareas

From t_departamento dep,  t_empleado emp, t_tareas_empleado tar

Where tar.fecha_tarea between to_date('01-01-2010','dd-mm-yyyy') and to_date('31-03-2010','dd-mm-yyyy')

    and emp.id_empleado = tar.id_empleado

    and dep.id_departamento = emp.id_departamento
```

```
Group by
    Dep.nombre_dep,
    Emp.nombre_emp,
    To_char(tar.fecha_tarea,'mm-yyyy')
Order by
    Dep.nombre_dep,
    Emp.nombre_emp,
    To_char(tar.fecha_tarea,'mm-yyyy');
```

Esta query está haciendo un GROUP BY por 3 campos, el nombre del departamento, nombre del empleado y mes-año de las tareas realizadas, fíjate que los campos del Select deben ser los mismos del Group By, solo se pueden agregar campos con funciones de grupo, en este caso solo se usa el Count, pero también se puede usar el Sum, Max, Min, etc. Además también se incluye el Order By por los mismos campos del Group by para que el resultado salga ordenado en tus Packages en Oracle.

Esta query entrega el siguiente resultado:

Departamento	Empleado	Mes-Año	Tareas
CONTABILIDAD	LUIS MARDONES	01-2010	365
CONTABILIDAD	LUIS MARDONES	02-2010	212
CONTABILIDAD	LUIS MARDONES	02-2010	645
CONTABILIDAD	MARIO SOLAR	01-2010	735
CONTABILIDAD	MARIO SOLAR	02-2010	235
CONTABILIDAD	MARIO SOLAR	03-2010	723
FINANZAS	MARCELO PEREZ	01-2010	435
FINANZAS	MARCELO PEREZ	02-2010	324
FINANZAS	MARCELO PEREZ	03-2010	575
FINANZAS	JUAN GALVEZ	01-2010	680

FINANZAS	JUAN GALVEZ	02-2010	380
FINANZAS	JUAN GALVEZ	03-2010	430

En este resultado solo se está mostrando parte del resultado, ya que solo se muestran 2 departamentos con dos empleados cada uno. Prueba este ejemplo con las tablas de tu Base de Datos para que comiences a soltar la mano practicando lo aprendido en tus Packages en Oracle.

Te propongo que pruebes de inmediato este ejemplo del GROUP BY en tu Base de Datos con tus propias tablas, procedimientos y funciones para que comiences a ejercitar la mano y aprendas a dominar esta nueva técnica.

Paso N° 3

Sepa Como Crear Campos Virtuales en las Querys de Oracle

Si aprendes como y cuando crear campos virtuales podrás mejorar la calidad de tus consultas Select a la Base de Datos Oracle, ya que un campo virtual te permite realizar muchas operaciones en tiempo de ejecución, por ejemplo, realizar cálculos, decodificar la columna de una tabla, ordenar los datos obtenidos, crear marcas o indicadores, etc.

Los campos virtuales son una herramienta de gran utilidad para la construcción de querys en Oracle, son muy fáciles de implementar y permiten mucha flexibilidad para ser usados dentro de un Select. Revisemos algunos ejemplos para que entiendas como se debe implementar un campo virtual.

Ejemplo #1: Supone que necesitas dos campos virtuales para contar y sumar los datos de una tabla, veamos cómo se hace:

```
/* Packages en Oracle */

Select

    Count(id_empleado) cantidad,

    Sum(dias_trab)  dias_trabajados

From t_empleado;
```

Este es un ejemplo típico donde se usan campos virtuales para contar y sumar registros, fíjate que además se incluye el nombre a cada campo para identificarlo (cantidad y dias_trabajados).

Ejemplo #2: Supone que necesitas crear un título que describa los datos y además ordenarlos por un criterio en particular, veamos cómo se hace:

```
/* Packages en Oracle */

Select

    Dat.titulo,

    Dat.orden,

    Dat.id_departamento,

    Dat.nombre_dep
```

```
From (

Select

    'DEPTO NUEVO' titulo,

    2 orden,

    Dep.id_departamento,

    Dep.nombre_dep

From t_departamento dep

Where dep.id_departamento = 101

Union all

Select

    'DEPTO ANTIGUO' titulo,

    Decode(dep.ind_estado_civil,'C',1,'S',2,3) orden,

    Dep.id_departamento,

    Dep.nombre_dep

From t_departamento dep

Where dep.id_departamento = 125

) dat

Order by dat.orden
```

En este ejemplo se crea el campo virtual Titulo para identificar los datos que pertenecen a cada departamento y el campo virtual Orden donde se indica un número el que permite ordenar finalmente la data, fíjate que el campo Orden de la segunda query contiene la función Decode para decodificar el campo ind_estado_civil y asignar un número para ordenar.

Te propongo que pruebes de inmediato estos ejemplos de Campos Virtuales en tu Base de Datos con tus propias tablas, procedimientos y funciones para que comiences a ejercitar la mano y aprendas a dominar esta nueva técnica.

Paso N° 4

Aprende a Eliminar los Registros Repetidos usando DISTINCT

Aprender a usar la cláusula DISTINCT te permitirá controlar los registros que estén repetidos en las consultas que realices a tu Base de Datos, ya que suele ocurrir que el id de una tabla padre se encuentre más de una vez en otra tabla hija, pero como resultado solo queremos mostrar los datos de la tabla padre.

Antes de tomar la decisión de usar la cláusula DISTINCT debes revisar muy bien tu query para determinar que los registros duplicados no se deban a que te falte o tengas mal construido un UNION en tu consulta lo que también suele ocurrir.

Revisemos un ejemplo práctico para entender mejor esta cláusula:

Ejemplo #1: Supongamos que tienes una query de dos tablas enlazadas entre sí por su campo clave, pero al ejecutar la query entrega filas duplicadas y solo se quiere que se muestren filas únicas, veamos cómo se hace:

```
Select dep.codigo_dep, dep.nombre_dep

From t_empleado emp, t_departamento dep

Where emp.id_departamento = dep.id_departamento

    And emp.ind_estado = 1 — 1=activo

Order by dep.nombre_dep;
```

Esta query permite listar todos los departamentos que tienen empleados activos ordenados por el nombre del departamento, pero si los departamentos tienen más de un empleado los departamentos se listarán duplicados o se repetirán por cada empleado activo del departamento, quedando el listado asi:

20 ADQUISICIONES

20 ADQUISICIONES

08 CONTABILIDAD

22 FINANZAS

15 INFORMATICA

15 INFORMATICA

15 INFORMATICA

15 INFORMATICA

35 RECURSOS HUMANOS

35 RECURSOS HUMANOS

35 RECURSOS HUMANOS

Los departamentos de Adquisiciones, Informática y Recursos Humanos se muestran repetidos por cada empleado activo que tiene cada uno, para solucionar este problema podemos echar mano de la cláusula DISTINCT de la siguiente manera:

Select **DISTINCT** dep.codigo_dep, dep.nombre_dep

From t_empleado emp, t_departamento dep

Where emp.id_departamento = dep.id_departamento

And emp.ind_estado = 1 – 1=activo

Order by dep.nombre_dep;

Incluyendo la cláusula DISTINCT en la query eliminamos todos los registros que estén repetidos en el resultado, quedando el listado asi:

20 ADQUISICIONES

08 CONTABILIDAD

22 FINANZAS

15 INFORMATICA

35 RECURSOS HUMANOS

Te propongo que pruebes de inmediato este ejemplo del DISTINCT en tu Base de Datos con tus propias tablas, procedimientos y funciones para que comiences a ejercitar la mano y aprendas a dominar esta nueva técnica.

Paso N° 5

Como Utilizar la Cláusula UNION en tus Consultas en Oracle

Si sabes cómo se debe usar la cláusula UNION del comando SELECT en tus consultas a la Base de Datos Oracle, podrás construir en una sola query todos los datos que obtengas de distintas fuentes de datos. Muchas veces ocurre que en una consulta SELECT en los Procedimientos en Oracle se deben traer datos de distintas fuentes pero que están relacionados de una u otra forma.

Muchos desarrolladores optan por construir querys separadas para traer los datos de cada una de las fuentes, pero esa forma de construir no es la más óptima ya que te tomará mucho más tiempo hacer las cosas por separado resultando un proceso ineficiente.

Para solucionar este problema Oracle provee la cláusula UNION en el comando SELECT la que permite unir una o varias querys en una sola consulta a la Base de Datos lo que resulta ser muy eficiente ya que solo se debe procesar una sola vez la consulta.

La cláusula UNION por si sola permite unificar los datos al unir las querys, es decir, si la primera query trae una fila de datos que también existe en la segunda query solo se muestra la primera y no ambas, evitando así que se muestren datos repetidos. UNION también cuenta con la variante ALL la que realiza el efecto contrario de usar solo UNION, es decir, permite mostrar todas las filas de todas las querys, aunque se encuentren duplicadas en tus Procedimientos en Oracle. Revisemos un ejemplo que permita entender mejor la cláusula UNION.

Ejemplo #1: Supone que debes mostrar todos los empleados de un departamento que tengan de 21 hasta 25 años que sean solteros, que tengan 26 hasta 30 años que estén casados y los que tengan más 30 que estén solteros o casados. Si te das cuenta esta consulta no se puede hacer en una sola query ya que se deben realizar filtros distintos para cada una, veamos cómo se hace esta consulta:

```
/* Procedimientos en Oracle */

Select

    Dep.nombre_dep,

    Emp.nombre_emp,

    Emp.edad,

    Decode(Emp.estado_civil,1,'Soltero',2,'Casado') estado_civil

From tb_empleado emp, t_departamento dep
```

```
Where dep.id_departamento = pin_id_departamento
    And emp.id_empleado = dep.id_empleado
    And emp.edad between 21 and 25
    And emp.estado_civil = 1 –1= soltero
```

UNION – Procedimientos en Oracle

```
Select
    Dep.nombre_dep,
    Emp.nombre_emp,
    Emp.edad,
    Decode(Emp.estado_civil,1,'Soltero',2,'Casado') estado_civil
From tb_empleado emp, t_departamento dep
Where dep.id_departamento = pin_id_departamento
    And emp.id_empleado = dep.id_empleado
    And emp.edad between 26 and 30
    And emp.estado_civil = 2 –2= Casado
```

UNION ALL – Procedimientos en Oracle

```
Select
    Dep.nombre_dep,
    Emp.nombre_emp,
    Emp.edad,
    Decode(Emp.estado_civil,1,'Soltero',2,'Casado') estado_civil
From tb_empleado emp, t_departamento dep
Where dep.id_departamento = pin_id_departamento
```

```
And emp.id_empleado = dep.id_empleado

And emp.edad > 30

And emp.estado_civil in (1,2) –1= Soltero, 2= Casado;
```

En este ejemplo se están uniendo 3 querys con la cláusula UNION, cada query tiene filtros diferentes, pero están mostrando los mismos campos como salida, para que las querys se unan correctamente deben tener los mismos campos de salida en el mismo orden y del mismo tipo de dato, de lo contrario Oracle generará un error al compilar o en tiempo de ejecución. Fíjate que en la última query use el UNION ALL para indicar que se muestren todas las filas de la consulta, aunque existan en las querys anteriores.

Te propongo que pruebes de inmediato este ejemplo del UNION en tu Base de Datos con tus propias tablas, procedimientos y funciones para que comiences a ejercitar la mano y aprendas a dominar esta nueva técnica.

Paso N° 6

Sepa Usar la Cláusula EXISTS del SELECT en los Packages en Oracle

Si sabes cómo y cuándo utilizar correctamente la cláusula EXISTS dentro de tus consultas SELECT podrás resolver problemas referentes a inconsistencias de datos que ocurran en tus procesos. Estas inconsistencias se pueden deber a que en una query se muestran registros que no corresponden debido a que ya se muestran en otra query.

La cláusula EXISTS siempre evalúa una subconsulta y devuelve un valor lógico el cual es verdadero (TRUE) si la subconsulta puesta dentro del EXISTS retorna filas y en caso contrario, es decir, la subconsulta no devuelve filas se devuelve un valor falso (FALSE). En el SELECT la cláusula EXISTS se puede consultar de forma positiva colocando solo la palabra EXISTS o de forma negativa agregando la palabra NOT quedando la consulta como NOT EXISTS.

Puedes pensar que EXISTS es muy similar a la cláusula IN pero EXISTS es mucho más rápida. Para entender mejor como se debe trabajar con la cláusula EXISTS veamos un ejemplo práctico.

Ejemplo #1: Supone que tienes una consulta por departamento a la tabla de empleados la que está mostrando filas que no corresponden debido a que algunos empleados se encuentran en proceso de evaluación.

```
/* Packages en Oracle*/

Select emp.rut, emp.dv, emp.nombre_emp

from t_empleado emp

where emp.id_departamento = pin_id_departamento

    and ind_estado = 9; — 9= ascendidos
```

Este ejemplo trae todos los empleados de un departamento en particular que tengan el estado de ascendido, pero no se deben mostrar empleados que están en evaluación, para solucionar este problema se debe usar la cláusula EXISTS, veamos cómo se debe hacer..

```
/* Packages en Oracle*/

Select emp.rut, emp.dv, emp.nombre_emp

from t_empleado emp
```

where emp.id_departamento = pin_id_departamento

and ind_estado = 9 — 9= ascendidos

and NOT EXISTS (select 1 from t_evaluacion eva where eva.id_empleado =

emp.id_empleado);

En este ejemplo se modifica la misma query anterior incluyendo la cláusula NOT EXISTS para que en la consulta no se incluyan los empleados que están en proceso de evaluación. Nota que en la subconsulta no es necesario seleccionar un campo para el Select, es suficiente con poner un texto, número o null, además fíjate que a la subconsulta en la condición del Where se está pasando como parámetro el id_empleado de la tabla t_empleado (emp.id_empleado).

Te propongo que pruebes de inmediato este ejemplo del EXISTS en tu Base de Datos con tus propias tablas, procedimientos y funciones para que comiences a ejercitar la mano y aprendas a dominar esta nueva técnica.

Paso N° 7

Aprende a Usar la Cláusula IN en los Procedimientos en Oracle

Saber usar eficientemente la cláusula IN te permitirá reducir la cantidad de código que escribas y además tener un código más legible que sea entendible por cualquier desarrollador en tus Procedimientos en Oracle.

La cláusula IN permite agrupar en un solo lugar dos o más códigos que hacen referencia a un mismo tipo de dato, eliminando así la cláusula OR en las condiciones del WHERE, del IF, del CASE, del WHILE, entre otros.

Revisemos algunos ejemplos prácticos que te permitan entender mejor como usar la cláusula IN.

Ejemplo #1: Supone que debes construir una condición en un IF donde una variable puede tomar varios valores diferentes, veamos la forma tradicional y la forma usando el IN:

Forma Tradicional.

```
/* Procedimientos en Oracle */

If lst_estado_civil = 'SOLTERO' or lst_estado_civil = 'SEPARADO' or lst_estado_civil = 'VIUDO' then

    <acción a realizar>

End if;
```

En este ejemplo se está usando el OR para separar cada valor que puede tomar la variable lst_estado_civil.

Forma usando el IN.

```
/* Procedimientos en Oracle */

If lst_estado_civil IN ('SOLTERO', 'SEPARADO', 'VIUDO') then

    <acción a realizar>

End if;
```

En este ejemplo se reemplaza el OR por un IN agrupando los distintos valores dentro de un paréntesis, lo que resulta ser más eficiente y ordenado.

Ejemplo #2: Veamos cómo se usa el IN en una consulta SELECT.

Forma Tradicional.

```
/* Procedimientos en Oracle */

Select emp.nombre_emp, emp.rut, emp.dv from t_empleado emp

where emp.id_estado_civil = 1  -— SOLTERO

    or emp.estado_civil = 3  -— SEPARADO

    or emp.estado_civil = 4;  -— VIUDO
```

En este ejemplo se está usando un OR para cada valor del campo emp.estado_civil.

Forma usando el IN.

```
/* Procedimientos en Oracle */

Select emp.nombre_emp, emp.rut, emp.dv from t_empleado emp where
emp.id_estado_civil IN (    1, — SOLTERO

   3, — SEPARADO

   4 — VIUDO

 );
```

En este ejemplo se usa el IN para reemplazar el OR.

En un SELECT también puedes hacer esto:

```
/* Procedimientos en Oracle */

Select emp.nombre_emp, emp.rut, emp.dv
from t_empleado emp
where emp.id_estado_civil IN (select civ.id_estado_civil from t_estado_civil civ where
civ.nom_estado_civil <> 'CASADO');
```

En este ejemplo se usa el IN obteniendo los id_estado_civil desde otra consulta Select.

Te propongo que pruebes de inmediato estos ejemplos de la cláusula IN en tu Base de Datos con tus propias tablas, procedimientos y funciones para que comiences a ejercitar la mano y aprendas a dominar esta nueva técnica.

Recomendación final

Para finalizar, me gustaría indicarte que todos estos pasos o técnicas que te he mostrado pueden parecer difíciles de aplicar y de hecho varias de ellas lo son, pero son los fundamentos básicos que debes saber y es primordial que los pongas en práctica cada vez que debas construir un Select en Oracle.

No debes desanimarte si al principio te cuesta entender estos pasos o no sepas como llevarlos a la práctica en un problema real que debas resolver, pero si te esfuerzas día a día y los practicas una y otra vez, te darás cuenta que será como andar en bicicleta; una vez que lo aprendes nunca más se olvida.

Algunas personas leen este material, pero nunca lo ponen en acción. Esto es inútil para ellos, y solo pierden su tiempo consumiendo información que nunca llevan a la práctica y por lo tanto nunca se van a superar a si mismos.

Por eso es fundamental llevar la teoría a la práctica y te propongo que tomes acción inmediatamente en tu Base de Datos Oracle y practiques cada una de las técnicas que te he enseñado en este reporte, para que comiences a soltar la mano y poco a poco ir construyendo instrucciones Select que sean eficientes.

Sinceramente,

Roberto Vicencio – Para Tu Éxito con Oracle
Director de DominaTusPackagesEnOracle.com

PD: Si deseas rápidamente ***Dominar el Lenguaje PL/SQL***, te invito a ver un ***Webinar Gratuito*** que estoy entregando a los alumnos con un ***Entrenamiento Avanzado*** de mucho valor sobre el ***SELECT*** y ***PL/SQL*** que te encantará, anótate haciendo clic en el siguiente enlace:

Descubre y Anótate a Este Webinar Gratis Haciendo Clic Aquí >>