tipos de datos sql

char: cadena de caracteres de longitud fija (rellena con

espacios en blanco si se ponen menos caracteres),

por defecto es 1 byte

varchar2: cadena de caracteres de longitud variable,

tamaño maximo de n bytes, obligatorio asignar un tamaño

number: numerico, no es obligatorio especificar tamaño

date: tipo fecha, desde -4712 hasta el 9999

long: para albergar caracteres de longitud variable,

hasta 2gb, se recomienda no usar

clob: alberga caracteres multi-byte o single-byte

se usa para textos grandes. (aun se usa)

nclob: igual que clob pero con caracteres unicode,

almacena cadenas de caracteres de tipo nacional de

distintos lenguajes

raw: datos binarios, mismos limites que varchar2

blob: objeto binario hasta 4gb

bfile: contiene punteros a ficheros externos de bd,

se guardan afuera de la bd

rowid: representa el id unico de una fila

timestamp: fecha que almacena fracciones de segundo,

hay variaciones como with timezone o with

localtimezone

interval year to month: almacena tiempo como intervalo

de años y meses

interval day to second: almacena intervalo de dias

horas minutos y segundos (mas especifico que el

anterior)

comando select

este permite realizar consultas (query) y extraer

datos de la base de datos

se pueden hacer tres operaciones basicas con este

comando, segun el modelo relacional

proyección: recuperar columnas de una tabla

seleccion: recuperar filas de una tabla

joins: recuperar informacion de manera conjunta

de varias tablas.

proyeccion, seleccionar algunas columnas

seleccion, seleccionar algunas filas.

SELECT DATOS FROM TABLA

-----------------------------------------------------

primeros pasos select.

es necesario poner ; al final de las sentencias para poder

ejecutarlas ordenadamente cuando existan varias en una hoja

de trabajo.

Bases de datos oracle, probando las funcionalidades.

LENGTH: muestra la cantidad de caracteres de los valores en una tabla

SUBSTR: devuelve caracteres de una cadena, ej:

select substr(first\_name,3,3); (del nombre carlos devolverá ‘los’, el primer 3 significa la posición desde donde comienza, y el segundo número significa cuantos caracteres recogerá desde ahí)

INSTR: Nos marca la posición numérica en donde se encuentra un carácter que asignamos para buscar, además se puede especificar desde qué posición de la cadena queremos buscar, ejemplo: para el nombre Catalina de la tabla pnombre\_emp

INSTR(lower(pnombre\_emp),'a',3) from empleado; = 4 (usamos lower para que todos los caracteres sean minúscula, buscamos el carácter ‘a’ desde la tercera posición de la cadena, entonces marcara el primero que encuentre desde ahí, por eso el resultado es 4)

distinc: este afecta a toda la fila (todas las columnas) y se pone despues del "select", genera resultados sin repetir valores.

Ejemplo DISTINCT: select distinct job\_id, department\_id from employees;

(esto evitaria repetir los id en los resultados)

WHERE: pone una condicion en la base de datos para poder filtrar las búsquedas, se usa despues de nombrar la tabla en donde se hace la busqueda.

EJEMPLO WHERE: select \* from employees where department\_id = 50;

(esto mostraria todas las tablas de empleados que tengan el department\_id 50)

EJEMPLO WHERE CON FECHAS: SELECT \* FROM EMPLOYEES WHERE hire\_date='21-06-2007';

(en este caso se pueden poner fechas entre '' y oracle las reconocerá)

where puede usar operadores aritmeticos (>, =, <, <>)

BETWEEN: Nos sirve para ver resultados entre los parametros que asignamos, se pone después del where.

EJEMPLO BETWEEN: SELECT \* FROM EMPLOYEES

WHERE SALARY BETWEEN 5000 AND 6000;

(muestra todos los empleados con salarios entre 5000 y 6000)

IN: Esta funcion permite poner varias cualidades que queremos que se cumplan, por ejemplo empleados del departamento 50 y 60, el in se pone despues del where y la condicion va entre ()

EJEMPLO IN: SELECT \* FROM EMPLOYEES WHERE department\_id IN (50,60);

(muestra empleados del departamento 50 y 60)

SELECT \* FROM EMPLOYEES WHERE job\_id IN ('SH\_CLERK','ST\_CLERK','ST\_MAN');

(muestra empleados que trabajan en el job que cumpla

los nombres mencionados)

LIKE: Funciona solo con los caracteres, sirve para buscar nombres que coincidan en la bd, podemos buscar por palabras que comienzen y terminen de una manera, que despues de n caracteres contenga la letra que buscamos etc

EJEMPLOS LIKE:

SELECT \* FROM EMPLOYEES WHERE

first\_name LIKE 'J%';

(Busca todos los nombres que comienzen con J)

SELECT \* FROM EMPLOYEES WHERE

first\_name LIKE 'S%n';

(Busca los nombres que Comienzen con S y terminen en n, sin importar los caracteres de en medio)

SELECT \* FROM COUNTRIES WHERE

country\_name LIKE '\_r%';

(selecciona los paises con una r en el segundo caracter de su nombre)

NULOS: Es un espacio vacio, sin ningún valor asignado

EJEMPLO NULL: Select \* from employees

WHERE commission\_pct IS NULL;

(busca valores nulos)

Select \* from employees

WHERE commission\_pct IS NOT NULL;

(busca valores que no son nulos)

(Mostrara empleados sin comission\_pct. no se pueden usar operadores aritmeticos porque no se pueden comparar valores nulos o inexistentes)

AND OR: SE PONEN DESPUES DEL WHERE Y SON CONDICIONES QUE PONEMOS PARA QUE SE CUMPLAN Y FILTREN LA BUSQUEDA

COMO USAR AND OR:

C1 AND C2 TRUE TRUE --> TRUE

C1 AND C2 TRUE FALSE --> FALSE

C1 AND C1 FALSE FALSE --> FALSE

C1 OR C2 TRUE TRUE --> TRUE

C1 OR C2 TRUE FALSE --> TRUE

C1 OR C2 FALSE FALSE --> FALSE

EJEMPLO DE AND OR:

SELECT first\_name, hire\_date, job\_id, salary FROM EMPLOYEES

WHERE job\_id = 'IT\_PROG' AND salary < 6000;

(ELEGIMOS LOS EMPLEADOS QUE TENGA EL TRABAJO IT\_PROG Y UN SALARIO MENOR A 6000)

ORDER BY: Sirve para ordenar como queremos ver los resultados, por defecto los resultados van ascendiendo, si se quiere descendente se debe especificar

EJEMPLO ORDER BY: SELECT first\_name, last\_name, salary FROM EMPLOYEES

WHERE first\_name ='David'

ORDER BY first\_name, salary;

(se ordena primero por nombre, y cuando se repite el nombre se ordena por el salario)

CLÁUSULA FETCH: puede limitar la cantidad de filas, se pone al final de las clausulas select

EJEMPLO FETCH: SELECT FIRST\_NAME, SALARY FROM EMPLOYEES FETCH FIRST 10 ROWS ONLY

(muestra las primeras 10 filas)

select first\_name,salary from employees

order by salary desc fetch first 7 rows with ties;

select first\_name,salary from employees

order by salary desc offset 5 rows fetch first 7 rows with ties;

-- calcula primer 20% de filas

select \* from employees fetch first 20 percent rows only;

—---------------------------------------------------------------------

ejemplos trabajados

select distinct city, state\_province from locations;

select distinct job\_id, department\_id from employees;

select \* from employees

where department\_id = 50;

/\* = > | < > | =| <=| <>

\*/

select first\_name,salary from employees

WHERE salary > 5000;

SELECT first\_name FROM EMPLOYEES

WHERE department\_id <> 50;

SELECT first\_name FROM EMPLOYEES WHERE first\_name<>'John';

SELECT \* FROM EMPLOYEES; --1 DE ENERO 4712 AC

SELECT \* FROM EMPLOYEES WHERE hire\_date>'21-06-2007';

-- PRACTICAS CON WHERE

SELECT \* FROM EMPLOYEES

WHERE department\_id = 100;

SELECT city, street\_address, country\_id FROM LOCATIONS

WHERE country\_id ='US';

SELECT \* FROM countries

WHERE region\_id = 3;

SELECT \* FROM EMPLOYEES

WHERE manager\_id <> 114;

SELECT \* FROM EMPLOYEES

WHERE hire\_date >= '1-1-2006';

SELECT \* FROM EMPLOYEES

WHERE job\_id = 'ST\_CLERK';

SELECT \* FROM EMPLOYEES

WHERE last\_name = 'Smith';

-- BETWEEN V1 AND V2

SELECT \* FROM EMPLOYEES

WHERE SALARY BETWEEN 5000 AND 6000;

SELECT \* FROM EMPLOYEES

WHERE hire\_date BETWEEN '01-01-2007' AND '01-01-2009';

SELECT \* FROM EMPLOYEES

WHERE first\_name BETWEEN 'Douglas' AND 'Steven';

-- IN (V1,V2,V3) 'SI EL VALOR ESTA DENTRO DE LA LISTA SE MUESTRA'

SELECT \* FROM EMPLOYEES WHERE department\_id IN (50,60);

SELECT \* FROM EMPLOYEES WHERE job\_id IN ('SH\_CLERK','ST\_CLERK','ST\_MAN');

-- PRACTICAS BETWEEN IN

SELECT \* FROM EMPLOYEES

WHERE department\_id IN (40,60);

SELECT \* FROM EMPLOYEES

WHERE hire\_date BETWEEN '01-01-2002' AND '31-12-2004';

SELECT \* FROM EMPLOYEES

WHERE last\_name BETWEEN 'D' AND 'G';

SELECT \* FROM EMPLOYEES

WHERE department\_id IN (30,60,90);

SELECT \* FROM EMPLOYEES

WHERE job\_id IN ('IT\_PROG','PU\_CLERK');

SELECT \* FROM LOCATIONS

WHERE country\_id IN ('UK','JP');

-- LIKE funciona solo con 'PATRON O CADENA DE CARACTERES'

-- %

-- \_

SELECT \* FROM EMPLOYEES

WHERE first\_name LIKE 'J%';

SELECT \* FROM EMPLOYEES WHERE

first\_name LIKE '\_e%';

SELECT \* FROM EMPLOYEES WHERE

first\_name LIKE '%te%';

-- PRACTICAS CON LIKE

/\*

Indicar los datos de los empleados cuyo FIRST\_NAME empieza por ‘J’

Averiguar los empleados que comienzan por ‘S’ y terminan en ‘n’

Indicar los países que tienen una “r” en la segunda letra (Tabla COUNTRIES)

\*/

SELECT \* FROM EMPLOYEES WHERE

first\_name LIKE 'J%';

SELECT \* FROM EMPLOYEES WHERE

first\_name LIKE 'S%n';

SELECT \* FROM COUNTRIES WHERE

country\_name LIKE '\_r%';

-- Nulos

Select \* from employees

WHERE commission\_pct IS NULL;

Select \* from employees

WHERE commission\_pct IS NOT NULL;

-- Practicas NULL

/\*

1 Listar las ciudades de la tabla LOCATIONS que no tienen STATE\_PROVINCE

2 Averiguar el nombre, salario y comisión de aquellos empleados que tienen

comisión.

3 También debemos visualizar una columna calculada denominada

“Sueldo Total”, que sea el sueldo más la comisión

\*/

Select \* from LOCATIONS

WHERE state\_province IS NULL ;

Select first\_name, salary, commission\_pct, (salary \* commission\_pct) + salary as "salario total" from employees

WHERE commission\_pct IS NOT NULL;

/\* CONDICIONES COMPLEJAS CON OPERADORES

AND C1 AND C2 --> TRUE SI 2 SE CUMPLEN

OR C1 OR C2 --> TRUE SI 1 SE CUMPLE

NOT C1 --> TRUE (NEGACION DEL RESULTADO)

\*/

SELECT \* FROM EMPLOYEES

WHERE salary > 5000 AND department\_id = 50;

SELECT \* FROM EMPLOYEES

WHERE salary > 5000 OR department\_id = 50;

SELECT \* FROM EMPLOYEES

WHERE salary > 5000 AND department\_id = 50;

SELECT \* FROM EMPLOYEES

WHERE department\_id NOT IN (50,60);

/\*

C1 AND C2 TRUE TRUE --> TRUE

C1 AND C2 TRUE FALSE --> FALSE

C1 AND C1 FALSE FALSE --> FALSE

C1 OR C2 TRUE TRUE --> TRUE

C1 OR C2 TRUE FALSE --> TRUE

C1 OR C2 FALSE FALSE --> FALSE

\*/

/\*

1 Obtener el nombre y la fecha de la entrada y el tipo de trabajo de los

empleados que sean IT\_PROG y que ganen menos de 6000 dólares

2 Seleccionar los empleados que trabajen en el departamento 50 o 80,

cuyo nombre comience por S y que ganen más de 3000 dólares.

3 ¿Qué empleados de job\_id IT\_PROG tienen un prefijo 5 en el teléfono

y entraron en la empresa en el año 2007?

\*/

-- PRACTICAS AND OR

SELECT first\_name, hire\_date, job\_id, salary FROM EMPLOYEES

WHERE job\_id = 'IT\_PROG' AND salary < 6000;

SELECT \* FROM EMPLOYEES

WHERE first\_name LIKE 'S%' AND (department\_id =50 OR department\_id = 80)

AND salary > 3000;

SELECT \* FROM EMPLOYEES

WHERE job\_id = 'IT\_PROG' AND phone\_number LIKE '5%' AND hire\_date BETWEEN

'01-01-2007' AND '31-12-2007';

-- ORDER BY

SELECT \* FROM EMPLOYEES

ORDER BY salary DESC;

SELECT \* FROM EMPLOYEES

ORDER BY salary;

SELECT \* FROM EMPLOYEES

ORDER BY first\_name, last\_name;

SELECT first\_name, last\_name, salary FROM EMPLOYEES

WHERE first\_name ='David'

ORDER BY first\_name, salary;

-- SE PUEDE MENCIONAR LA COLUMNA A ORDENAR POR SU NOMBRE O POR SU POSICIÓN

-- O POR SU ALIAS

SELECT first\_name, salary\*12 FROM EMPLOYEES

ORDER BY SALARY\*12;

SELECT first\_name, salary\*12 FROM EMPLOYEES

ORDER BY 2;

SELECT first\_name, salary\*12 AS SALARIO FROM EMPLOYEES

ORDER BY SALARIO DESC;

--FETCH

SELECT first\_name, salary FROM EMPLOYEES FETCH FIRST 10 ROWS ONLY;

---------------------------------------------------------------------------

FUNCIONES

se pueden usar en cualquier parte de sql, despues de ellas se pone en paréntesis cuál será la columna afectada por la función

UPPER: convierte todos los caracteres a mayusculas

ejemplo upper:

LOWER: hace a todos los caracteres minusculas

INITCAP: pone en mayusculas la primera letra de cada palabra

LENGTH: MUESTRA EL LARGO DE UNA CADENA Y SE PUEDE USAR PARA FILTRAR

SUBSTR: EXTRAE los caracteres solicitados de una cadena

EJEMPLO SUBSTR: SELECT first\_name,SUBSTR(first\_name,1,4) FROM EMPLOYEES;

SELECT first\_name,SUBSTR(first\_name,3) FROM EMPLOYEES;

SELECT first\_name,SUBSTR(first\_name, length(first\_name),1) FROM EMPLOYEES

PRACTICAS HECHAS EN SQL DEVELOPER

select distinct city, state\_province from locations;

select distinct job\_id, department\_id from employees;

select \* from employees

where department\_id = 50;

/\*

=

>

<

>=

<=

<>

\*/

select first\_name,salary from employees

WHERE salary > 5000;

SELECT first\_name FROM EMPLOYEES

WHERE department\_id <> 50;

SELECT first\_name FROM EMPLOYEES WHERE first\_name<>'John';

SELECT \* FROM EMPLOYEES; --1 DE ENERO 4712 AC

SELECT \* FROM EMPLOYEES WHERE hire\_date>'21-06-2007';

-- PRACTICAS CON WHERE

SELECT \* FROM EMPLOYEES

WHERE department\_id = 100;

SELECT city, street\_address, country\_id FROM LOCATIONS

WHERE country\_id ='US';

SELECT \* FROM countries

WHERE region\_id = 3;

SELECT \* FROM EMPLOYEES

WHERE manager\_id <> 114;

SELECT \* FROM EMPLOYEES

WHERE hire\_date >= '1-1-2006';

SELECT \* FROM EMPLOYEES

WHERE job\_id = 'ST\_CLERK';

SELECT \* FROM EMPLOYEES

WHERE last\_name = 'Smith';

-- BETWEEN V1 AND V2

SELECT \* FROM EMPLOYEES

WHERE SALARY BETWEEN 5000 AND 6000;

SELECT \* FROM EMPLOYEES

WHERE hire\_date BETWEEN '01-01-2007' AND '01-01-2009';

SELECT \* FROM EMPLOYEES

WHERE first\_name BETWEEN 'Douglas' AND 'Steven';

-- IN (V1,V2,V3) 'SI EL VALOR ESTA DENTRO DE LA LISTA SE MUESTRA'

SELECT \* FROM EMPLOYEES WHERE department\_id IN (50,60);

SELECT \* FROM EMPLOYEES WHERE job\_id IN ('SH\_CLERK','ST\_CLERK','ST\_MAN');

-- PRACTICAS BETWEEN IN

SELECT \* FROM EMPLOYEES

WHERE department\_id IN (40,60);

SELECT \* FROM EMPLOYEES

WHERE hire\_date BETWEEN '01-01-2002' AND '31-12-2004';

SELECT \* FROM EMPLOYEES

WHERE last\_name BETWEEN 'D' AND 'G';

SELECT \* FROM EMPLOYEES

WHERE department\_id IN (30,60,90);

SELECT \* FROM EMPLOYEES

WHERE job\_id IN ('IT\_PROG','PU\_CLERK');

SELECT \* FROM LOCATIONS

WHERE country\_id IN ('UK','JP');

-- LIKE funciona solo con 'PATRON O CADENA DE CARACTERES'

-- %

-- \_

SELECT \* FROM EMPLOYEES

WHERE first\_name LIKE 'J%';

SELECT \* FROM EMPLOYEES WHERE

first\_name LIKE '\_e%';

SELECT \* FROM EMPLOYEES WHERE

first\_name LIKE '%te%';

-- PRACTICAS CON LIKE

/\*

Indicar los datos de los empleados cuyo FIRST\_NAME empieza por ‘J’

Averiguar los empleados que comienzan por ‘S’ y terminan en ‘n’

Indicar los países que tienen una “r” en la segunda letra (Tabla COUNTRIES)

\*/

SELECT \* FROM EMPLOYEES WHERE

first\_name LIKE 'J%';

SELECT \* FROM EMPLOYEES WHERE

first\_name LIKE 'S%n';

SELECT \* FROM COUNTRIES WHERE

country\_name LIKE '\_r%';

-- Nulos

Select \* from employees

WHERE commission\_pct IS NULL;

Select \* from employees

WHERE commission\_pct IS NOT NULL;

-- Practicas NULL

/\*

1 Listar las ciudades de la tabla LOCATIONS que no tienen STATE\_PROVINCE

2 Averiguar el nombre, salario y comisión de aquellos empleados que tienen

comisión.

3 También debemos visualizar una columna calculada denominada

“Sueldo Total”, que sea el sueldo más la comisión

\*/

Select \* from LOCATIONS

WHERE state\_province IS NULL ;

Select first\_name, salary, commission\_pct, (salary \* commission\_pct) + salary as "salario total" from employees

WHERE commission\_pct IS NOT NULL;

/\* CONDICIONES COMPLEJAS CON OPERADORES

AND C1 AND C2 --> TRUE SI 2 SE CUMPLEN

OR C1 OR C2 --> TRUE SI 1 SE CUMPLE

NOT C1 --> TRUE (NEGACION DEL RESULTADO)

\*/

SELECT \* FROM EMPLOYEES

WHERE salary > 5000 AND department\_id = 50;

SELECT \* FROM EMPLOYEES

WHERE salary > 5000 OR department\_id = 50;

SELECT \* FROM EMPLOYEES

WHERE salary > 5000 AND department\_id = 50;

SELECT \* FROM EMPLOYEES

WHERE department\_id NOT IN (50,60);

/\*

C1 AND C2 TRUE TRUE --> TRUE

C1 AND C2 TRUE FALSE --> FALSE

C1 AND C1 FALSE FALSE --> FALSE

C1 OR C2 TRUE TRUE --> TRUE

C1 OR C2 TRUE FALSE --> TRUE

C1 OR C2 FALSE FALSE --> FALSE

\*/

/\*

1 Obtener el nombre y la fecha de la entrada y el tipo de trabajo de los

empleados que sean IT\_PROG y que ganen menos de 6000 dólares

2 Seleccionar los empleados que trabajen en el departamento 50 o 80,

cuyo nombre comience por S y que ganen más de 3000 dólares.

3 ¿Qué empleados de job\_id IT\_PROG tienen un prefijo 5 en el teléfono

y entraron en la empresa en el año 2007?

\*/

-- PRACTICAS AND OR

SELECT first\_name, hire\_date, job\_id, salary FROM EMPLOYEES

WHERE job\_id = 'IT\_PROG' AND salary < 6000;

SELECT \* FROM EMPLOYEES

WHERE first\_name LIKE 'S%' AND (department\_id =50 OR department\_id = 80)

AND salary > 3000;

SELECT \* FROM EMPLOYEES

WHERE job\_id = 'IT\_PROG' AND phone\_number LIKE '5%' AND hire\_date BETWEEN

'01-01-2007' AND '31-12-2007';

-- ORDER BY

SELECT \* FROM EMPLOYEES

ORDER BY salary DESC;

SELECT \* FROM EMPLOYEES

ORDER BY salary;

SELECT \* FROM EMPLOYEES

ORDER BY first\_name, last\_name;

SELECT first\_name, last\_name, salary FROM EMPLOYEES

WHERE first\_name ='David'

ORDER BY first\_name, salary;

-- SE PUEDE MENCIONAR LA COLUMNA A ORDENAR POR SU NOMBRE O POR SU POSICIÓN

-- O POR SU ALIAS

SELECT first\_name, salary\*12 FROM EMPLOYEES

ORDER BY SALARY\*12;

SELECT first\_name, salary\*12 FROM EMPLOYEES

ORDER BY 2;

SELECT first\_name, salary\*12 AS SALARIO FROM EMPLOYEES

ORDER BY SALARIO DESC;

--FETCH

SELECT first\_name, salary FROM EMPLOYEES FETCH FIRST 10 ROWS ONLY;

/\*

En la tabla LOCATIONS, averiguar las ciudades que son de Canada o

Estados unidos (Country\_id=CA o US) y que la longitud del nombre de la

calle sea superior a 15.

• Muestra la longitud del nombre y el salario anual (por 14) para los

empleados cuyo apellido contenga el carácter 'b' después de la 3ª

posición.

• Averiguar los empleados que ganan entre 4000 y 7000 euros y que

tienen alguna 'a' en el nombre. (Debemos usar INSTR y da igual que sea

mayúscula que minúsculas) y que tengan comisión.

• Visualizar las iniciales de nombre y apellidos separados por puntos. Por

ejemplo; ellen abel = E.A

• Mostrar empleados donde el nombre o apellido comienza con S.. • Visualizar el nombre del empleado, su salario, y con asteriscos, el número miles de dólares que gana. Se asocia ejemplo. (PISTA: se puede usar RPAD. Ordenado por salario

\*/

SOLUCIONES

-- 1 ejercicio

select city,country\_id, street\_address from locations

where country\_id IN('CA','US') AND LENGTH(street\_address) > 15;

-- 2 ejercicio

SELECT first\_name||' '||last\_name AS NOMBRE,LENGTH(first\_name||' '||last\_name) AS LARGO,

salary\*14 SALARIO

FROM EMPLOYEES

WHERE INSTR(last\_name,'b') > 3;

-- 3 EJERCICIO

select \* from employees

WHERE (salary BETWEEN 4000 AND 7000) AND INSTR(LOWER(first\_name),'a')<> 0

AND commission\_pct IS NOT NULL;

-- 4 EJERCICIO

SELECT first\_name, last\_name, SUBSTR(first\_name,1,1)||'.'||SUBSTR(last\_name,1,1)||'.'

AS INICIALES FROM EMPLOYEES;

-- 5

SELECT first\_name, last\_name FROM EMPLOYEES

WHERE FIRST\_NAME LIKE('S%') OR last\_name LIKE('S%');

-- 6

SELECT first\_name, salary, RPAD('\*',salary/1000,'\*') AS RANKING FROM EMPLOYEES

ORDER BY salary DESC;

-- PRACTICAS NUMERICAS

/\*

• Visualizar el nombre y salario de los empleados de los que el número de

empleado es impar (PISTA: MOD)

• Prueba con los siguientes valores aplicando las funciones TRUNC y

ROUND, con 1 y 2 decimales.

\*/

SOLUCIONES

—---------------------

SELECT employee\_id,first\_name, salary FROM EMPLOYEES

WHERE MOD(employee\_id, 2) <> 0;

SELECT ROUND(25.67,0) FROM DUAL;

SELECT ROUND(25.67,1) FROM DUAL;

SELECT ROUND(25.34,1) FROM DUAL;

SELECT ROUND(25.34,2) FROM DUAL;

SELECT ROUND(25.67,-1) FROM DUAL;

SELECT TRUNC(25.67,0) FROM DUAL;

SELECT TRUNC(25.67,1) FROM DUAL;

SELECT TRUNC(25.34,1) FROM DUAL;

SELECT TRUNC(25.34,2) FROM DUAL;

SELECT TRUNC(25.67,-1) FROM DUAL;

—-----------------------------------------------

20/01/2022

--to\_char(date,'formato')

/\*

AM PM MERIDIAN

HH FORMATO 12 HORAS

HH24 FORMATO 24 HORAS

MI MINUTO

SS SEGUNDOS

\*/

SELECT SYSDATE, TO\_CHAR(SYSDATE,'HH') FROM DUAL;

SELECT SYSDATE, TO\_CHAR(SYSDATE,'HH24') FROM DUAL;

SELECT SYSDATE, TO\_CHAR(SYSDATE,'MI') FROM DUAL;

SELECT SYSDATE, TO\_CHAR(SYSDATE,'SS') FROM DUAL;

SELECT SYSDATE, TO\_CHAR(SYSDATE,'"SON LAS" HH24:MI "DEL DIA" DAY "en el año" yyyy') FROM DUAL;

1. Funciones de conversion. TO\_CHAR

• Indicar los empleados que entraron en Mayo en la empresa. Debemos

buscar por la abreviatura del mes

• Indicar los empleados que entraron en el año 2007 usando la función

to\_char

• ¿Qué día de la semana (en letra) era el día que naciste?

• Averiguar los empleados que entraron en el mes de Junio. Debemos

preguntar por el mes en letra. Nota: La función TO\_CHAR puede

devolver espacios a la derecha)

• Visualizar el salario de los empleados con dos decimales y en dólares y

también en la moneda local (el ejemplo es con euros, suponiendo que el

cambio esté en 0,79$)

\*/

SELECT \* FROM EMPLOYEES WHERE

TO\_CHAR(hire\_date,'MON') = 'MAY';

SELECT \* FROM EMPLOYEES

WHERE TO\_CHAR(hire\_date,'YYYY')= 2007;

SELECT TO\_CHAR(TO\_DATE('02-05-1998'),'DAY') FROM DUAL;

SELECT \* FROM EMPLOYEES WHERE

TO\_CHAR(hire\_date,'MON') = 'JUN';

SELECT SALARY, TO\_CHAR(salary,'$99999'), TO\_CHAR(salary,'L99999'), TO\_CHAR(salary,'99999.99') FROM EMPLOYEES;

--TO\_DATE(STRING,'FORMATO')

SELECT TO\_DATE('10-01-89') FROM DUAL;

SELECT TO\_DATE('10-01-1989') FROM DUAL;

SELECT TO\_DATE('10-JUN-89') FROM DUAL;

--FECHA SIN FORMATEAR FALLA

SELECT TO\_DATE('12-22-89') FROM DUAL;

-- FORMATEANDO LA FECHA PARA QUE SEPA COMO LEERLA

SELECT TO\_DATE('12-22-89','MM,DD,YY') FROM DUAL;

SELECT TO\_DATE('JUN-22-89','mm-dd-yy') FROM DUAL;

/\*

RR

yy

0-49 siglo actual

50-99 siglo 1900

USARA ESTE METODO AUTOMATICO CON RR

YY ES PARA PONER CUALQUIER FORMATO MANUALMENTE

\*/

SELECT TO\_CHAR(TO\_DATE('10-01-89'),'DD-MM-YYYY')FROM DUAL;

SELECT TO\_CHAR(TO\_DATE('10-01-39'),'DD-MM-YYYY')FROM DUAL;

SELECT TO\_CHAR(TO\_DATE('10-01-39','DD-MM-RR'),'DD-MM-YYYY') FROM DUAL;

SELECT TO\_CHAR(TO\_DATE('10-01-99','DD-MM-YY'),'DD-MM-YYYY') FROM DUAL;

--TO\_NUMBER(STRING,'FORMATO')

SELECT TO\_NUMBER('1000.89','9999.99') FROM DUAL;

SELECT TO\_NUMBER('$1000','L9999')FROM DUAL;

SELECT TO\_NUMBER('1210.73','9999.99') FROM DUAL;

SELECT TO\_NUMBER('$127.2','$9999.9') FROM DUAL;

SELECT TO\_NUMBER('$127.2','$99999') FROM DUAL;

select TO\_DATE('10-febrero-2018','dd-month-yyyy') from dual;

select phone\_number,TO\_NUMBER(SUBSTR(PHONE\_NUMBER,1,3))\*2 from employees;

--NULOS NVL(EXPRESION,VALOR) PUEDE DEVOLVER UN VALOR CONCRETO QUE EXPRESE LA CARACTERISTICA

SELECT NVL('HOLA','ADIOS') FROM DUAL;

SELECT NVL(NULL,'ADIOS') FROM DUAL;

SELECT NVL(NULL,NULL) FROM DUAL;

SELECT first\_name, commission\_pct\*salary, NVL(commission\_pct,0)\*SALARY FROM EMPLOYEES;

-- CON V SIN VALOR

-- NVL2(EXPRESION,VALOR1,VALOR2)

SELECT first\_name,SALARY,NVL2(commission\_pct,salary\*commission\_pct,salary\*0.1), commission\_pct FROM EMPLOYEES;

--NULLIF COMPARA DOS VALORES, SI SON IGUALES SALE NULO, SI NO SALE PRIMER VALOR (V1,V2)

SELECT NULLIF(1,1) FROM DUAL;

SELECT NULLIF(1,10) FROM DUAL;

-- SI EL RESULTADO ES IGUAL DEVUELVE NULO, SINO PRIMER VALOR

SELECT COUNTRY\_ID,UPPER(SUBSTR(COUNTRY\_NAME,1,2)), country\_name, NULLIF(country\_id,UPPER(SUBSTR(country\_name,1,2))),

NVL2( NULLIF(country\_id,UPPER(SUBSTR(country\_name,1,2))), 'NO SON IGUALES','SON IGUALES')

FROM COUNTRIES;

-- COALESCE(V1,V2,V3,V4) PERMITE PONER MUCHOS VALORES, CUANDO ENCUENTRA EL PRIMER NO NULO LO MUESTRA, SINO AVANZA

SELECT COALESCE(NULL,'VALOR1','VALOR2','VALOR3') FROM DUAL;

SELECT COALESCE(NULL,NULL,NULL,'VALOR3') FROM DUAL;

SELECT first\_name, TO\_CHAR(commission\_pct), TO\_CHAR(manager\_id) FROM EMPLOYEES;

SELECT first\_name, COALESCE(TO\_CHAR(commission\_pct), TO\_CHAR(manager\_id),'SIN JEFE NI COMISION') FROM EMPLOYEES;

-- EJERCICIOS DE PRACTICA

/\*

• De la tabla LOCATIONS visualizar el nombre de la ciudad y el estadoprovincia. En el caso de que no tenga que aparezca el texto “No tiene”

• Visualizar el salario de los empleados incrementado en la comisión

(PCT\_COMMISSION). Si no tiene comisión solo debe salir el salario

• Seleccionar el nombre del departamento y el manager\_id. Si no tiene,

debe salir un -1

• De la tabla LOCATIONS, devolver NULL si la ciudad y la provincia son

iguales. Si no son iguales devolver la CITY.

\*/

select city, NVL(state\_province,'NO TIENE') from locations;

SELECT salary,commission\_pct,NVL2(COMMISSION\_PCT,SALARY+SALARY\*COMMISSION\_PCT/100,SALARY) FROM EMPLOYEES;

SELECT department\_name,manager\_id,COALESCE(TO\_CHAR(manager\_id),'-1') FROM DEPARTMENTS;

SELECT department\_name, manager\_id,NVL(manager\_id,-1) FROM DEPARTMENTS;

SELECT city,state\_province, NULLIF(city,state\_province) FROM LOCATIONS;

select first\_name,job\_id,

CASE JOB\_ID

WHEN 'AD\_PRES' THEN 'TIPO 1'

WHEN 'SA\_REP' THEN 'TIPO 2'

WHEN UPPER('st\_clerk') THEN 'TIPO 3'

WHEN 'IT\_PROG' THEN 'TIPO 4'

WHEN 'SH\_CLERK' THEN 'TIPO 5'

else 'SIN TIPO'

END CASE

from employees

WHERE DEPARTMENT\_ID = 50;

-- CASE SEARCHED, SE PUEDEN USAR OPERADORES LOGICOS, CON MAYOR CAPACIDAD QUE EL ANTERIOR

SELECT first\_name, salary,

CASE

WHEN salary < 2500 OR SALARY > 23000 THEN 'ERES POBRE'

WHEN SALARY BETWEEN 2500 AND 3500 THEN 'CLASE MEDIA BAJA'

WHEN SALARY BETWEEN 3501 AND 4500 THEN 'CASI GERENTE'

WHEN SALARY BETWEEN 4501 AND 5000 THEN 'ESTAFADOR PROFESIONAL'

WHEN SALARY BETWEEN 5001 AND 15000 THEN 'MILLONARIO'

WHEN SALARY > 15000 THEN 'PIÑERA'

END

FROM EMPLOYEES

ORDER BY SALARY;

-- DECODE, MAS ESTANDAR Y ANTIGUA QUE EL CASE

SELECT first\_name, department\_id,

DECODE(department\_id,50,'informatica',90,'arquitecto',60,'matematico',80,'chofer','otro rol')

from employees;

/\*

1. Expresiones condicionales

• Visualizar los siguientes datos con CASE.

o Si el departamento es 50 ponemos Transporte

o Si el departamento es 90 ponemos Dirección

o Cualquier otro número ponemos “Otro departamento”

• Mostrar de la tabla LOCATIONS, la ciudad y el país. Ponemos los

siguientes datos dependiendo de COUNTRY\_ID.

o Si es US y CA ponemos América del Norte

o Si es CH, UK, DE,IT ponemos Europa

o Si es BR ponemos América del Sur

o Si no es ninguno ponemos ‘Otra zona’

• Realizar el primer ejercicio con DECODE en vez de con CASE

\*/

-- SOLUCIONES

-- EJERCICIO 1 CON DECODE

SELECT FIRST\_NAME, DEPARTMENT\_ID,

DECODE(department\_id,50,'TRANSPORTE',90,'DIRECCION','OTRO DEPARTAMENTO')

FROM EMPLOYEES;

-- EJERCICIO 1 CON CASE

select first\_name, department\_id,

CASE department\_id

WHEN 50 THEN 'TRANSPORTE'

WHEN 90 THEN 'DIRECCION'

ELSE 'OTRO DEPARTAMENTO'

END CASE

from employees;

-- EJERCICIO 2

SELECT city,country\_id,

CASE COUNTRY\_ID

WHEN 'US' THEN 'AMERICA DEL NORTE'

WHEN 'CA' THEN 'AMERICA DEL NORTE'

WHEN 'BR' THEN 'AMERICA DEL SUR'

ELSE 'OTRA ZONA'

END

FROM LOCATIONS;

-- VERSION 2 EJERCICIO 2

SELECT CITY,COUNTRY\_ID,

CASE

WHEN COUNTRY\_ID IN('US','CA') THEN 'AMERICA DEL NORTE'

WHEN COUNTRY\_ID = 'BR' THEN 'AMERICA DEL SUR'

ELSE 'OTRA ZONA'

END CASE

FROM LOCATIONS;

-- FUNCIONES DE GRUPO, ESTAS NO PUEDEN JUNTARSE CON FUNCIONES SIMPLES

--AVG CALCULA PROMEDIO

SELECT AVG(SALARY) FROM EMPLOYEES;

--MIN CALCULA MINIMO

SELECT MIN(SALARY) FROM EMPLOYEES;

--MAX CALCULA MAXIMO

SELECT MAX(SALARY) FROM EMPLOYEES;

SELECT AVG(SALARY), MIN(SALARY), MAX(SALARY) FROM EMPLOYEES

WHERE DEPARTMENT\_ID= 50;

SELECT MAX(HIRE\_DATE), MIN(HIRE\_DATE) FROM EMPLOYEES;

SELECT MAX(FIRST\_NAME), MIN(FIRST\_NAME) FROM EMPLOYEES;

-- count cuenta todos los datos aunque sean repetidos, pero no los nulos

-- hay 107 empleados de los que solo 35 tienen comision

select count(first\_name), count(commission\_pct) from employees;

select count(employee\_id) from employees;

select count(\*) from employees

where department\_id = 60;

select count(\*) from employees

where salary > 6000;

-- distinct muestra resultados sin repetir

select count(distinct first\_name) from employees;

select count(distinct department\_id) from employees;

select distinct department\_id from employees;

-- sum

select sum(salary) as "salario mensual",sum(salary)\*12 as "salario anual" from employees

where department\_id = 50;

select max(salary)-min(salary) from employees;

-- natural join

-- EN LOS NATURAL JOIN NO SE DEBE PONER ID CON COLUMNA PORQUE LA PONE

AUTOMATICA

SELECT re.region\_id, region\_name, COUNTRY\_ID, COUNTRY\_NAME

FROM

REGIONS RE NATURAL JOIN COUNTRIES CO;

-- LA TABLA DE MISMO NOMBRE CON DISTINTAS TABLAS SE PONE SIN SU ALIAS

-- EN NATURAL JOIN

-- ES PELIGROSO NATURAL JOIN PORQUE PUEDE UNIR COLUMNAS DE TABLAS DISTINTAS

-- CON EL MISMO NOMBRE PERO DISTINTO SIGNIFICADO, DANDO RESULTADOS INCOHERENTES

-- SOLO SIRVE CUANDO SE UNEN TABLAS CON LOS MISMOS DATOS Y MISMO NOMBRE

SELECT REGION\_NAME, COUNTRY\_NAME

FROM

REGIONS RE NATURAL JOIN COUNTRIES CO;

-- USING

-- NOS SIRVE PARA ESPECIFICAR CUAL SERA LA COLUMNA DE UNION QUE QUEREMOS USAR

SELECT DEPARTMENT\_NAME, FIRST\_NAME

FROM EMPLOYEES E JOIN DEPARTMENTS D

USING (DEPARTMENT\_ID)

WHERE SALARY > 5000;

-- ON con un and.(funciona como un where)

SELECT E.DEPARTMENT\_ID ,D.DEPARTMENT\_NAME, E.FIRST\_NAME,

L.CITY

FROM EMPLOYEES E JOIN DEPARTMENTS D

ON (E.DEPARTMENT\_ID=d.department\_id)

JOIN LOCATIONS L

ON (d.location\_id=l.location\_id)

and SALARY > 5000;

-- PRACTICAS JOIN ON

select country\_name, region\_name from countries

natural join

-- GROUP BY, SE PUEDE UTILIZAR CON FUNCIONES DE GRUPO

-- JUNTO A TODAS LAS COLUMNAS AFECTADAS POR UN GROUP BY

select department\_id, sum(salary),max(salary),trunc(avg(salary)) from employees

group by department\_id

ORDER BY DEPARTMENT\_ID;

-- HAVING ES COMO EL WHERE PERO PARA FUNCIONES DE GRUPO

SELECT DEPARTMENT\_ID, JOB\_ID, COUNT(\*), SUM(SALARY)

FROM EMPLOYEES

GROUP BY DEPARTMENT\_ID, JOB\_ID

HAVING SUM (SALARY) > 25000 AND COUNT(\*) > 10

ORDER BY DEPARTMENT\_ID;

-- PRACTICAS FUNCIONES DE GRUPO

SELECT DEPARTMENT\_ID,COUNT(\*) FROM EMPLOYEES

WHERE department\_id = 50

GROUP BY DEPARTMENT\_ID;

SELECT count(\*) FROM EMPLOYEES

where hire\_date BETWEEN '01-01-2007' and '31-12-2007';

select max(salary)-min(salary) from employees;

select sum(salary) FROM employees

where department\_id = 100;

select department\_id,ROUND(avg(salary),2) from employees

GROUP BY DEPARTMENT\_ID;

select country\_id, count(city) from countries c natural join

locations

GROUP by country\_id

ORDER BY COUNT(CITY) DESC;

SELECT DEPARTMENT\_ID, round(AVG(SALARY),2) FROM EMPLOYEES

WHERE COMMISSION\_PCT IS NOT NULL GROUP BY DEPARTMENT\_ID;

select department\_id,trunc(avg(salary))

, sum(commission\_pct) as sas from employees

where commission\_pct is not null

group by department\_id;

-- TO\_CHAR(hire\_date,'YYYY') MOSTRAR AÑOS CON MAS DE 10 EMPLEADOS

select TO\_CHAR(hire\_date,'YYYY') AÑO, COUNT(\*)

FROM EMPLOYEES

GROUP BY TO\_CHAR(HIRE\_DATE,'YYYY')

HAVING COUNT (\*) > 10

ORDER BY AÑO;

SELECT DEPARTMENT\_ID, TO\_CHAR(HIRE\_DATE,'YYYY') AÑO, COUNT(\*)

FROM EMPLOYEES

GROUP BY DEPARTMENT\_ID, TO\_CHAR(HIRE\_DATE,'YYYY')

ORDER BY DEPARTMENT\_ID;

SELECT DISTINCT DEPARTMENT\_ID, COUNT(employee\_id), COUNT(manager\_id) MANAGER

FROM EMPLOYEES

GROUP BY DEPARTMENT\_ID, MANAGER\_ID

HAVING COUNT(manager\_id) > 5;

-- practica joins

-- natural join para ver nombre y pais de region

select country\_name, region\_name from countries

natural join regions;

-- usando ejemplo anterior ver tambien ciudad

select c.country\_name, region\_name, city

from countries c join locations l on (c.country\_id=l.country\_id)

join regions r on (c.region\_id=r.region\_id);

-- indicar nombre de departamento y la media de sus salarios

select department\_name, round(avg(salary),2)

from employees e join departments d on (e.department\_id=d.department\_id)

group by d.department\_name;

-- mismo con using

select department\_name, round(avg(salary),2)

from employees e join departments d using (department\_id)

group by d.department\_name;

-- nombre departamento manager y ciudad

select d.department\_name,e.first\_name, city from departments d

join employees e on (d.manager\_id=e.employee\_id)

join locations l on (l.location\_id=d.location\_id);

-- job title

select j.job\_title, d.department\_name, e.last\_name,

e.hire\_date as inicio from jobs j

join employees e on (e.job\_id=j.job\_id)

join departments d on (e.department\_id=d.department\_id)

where e.hire\_date > '01-01-2000' and e.hire\_date < '01-01-2004';

-- lo mismo con using

select last\_name, hire\_date, department\_name, job\_title

from employees join departments

using (department\_id)

join jobs

using (job\_id)

where to\_char (hire\_date,'yyyy') between 2002 and 2004;

-- job title y media salarios superiores a 7000

select j.job\_title, avg(e.salary) salario from jobs j join

employees e on (e.job\_id=j.job\_id)

group by j.job\_title

having avg(e.salary) > 7000;

-- region y num departamento

SELECT REGION\_NAME,COUNT(\*) AS "NUM DEPAR"

FROM REGIONS NATURAL JOIN COUNTRIES NATURAL JOIN LOCATIONS

NATURAL JOIN DEPARTMENTS

GROUP BY REGION\_NAME;

-- mostrar nombre de empleado, departamento y país

select e.first\_name, d.department\_name,c.country\_name

from employees e join departments d on d.department\_id=e.department\_id

join locations l on l.location\_id=d.location\_id join

countries c on c.country\_id=l.country\_id;

-- mismo con using

select e.first\_name, d.department\_name,c.country\_name

from employees e join departments d using

(department\_id)

join locations l USING (LOCATION\_ID) join

countries c USING (COUNTRY\_ID);

-- cursores pl/sql

/\*

implicitos

explicitos

los cursores explicitos se abren y se cierran, son mas potentes que los implicitos,

pero eso no significa que se deban usar en todos los casos

\*/

/\*

SQL%ISOPEN -- SOLO VALE PARA CURSORES EXPLÍCITOS ¿ESTA ABIERTO EL CURSOR?

SQL%FOUND -- ¿SE ENCUENTRA UN CURSOR? SOLO APLICA A LO QUE ESTA ANTES DE LANZARLO

SQL%NOTFOUND-- ¿NO SE ENCUENTRA?

SQL%ROWCOUNT -- CUENTA LAS FILAS DEL CURSOR

\*/

begin

UPDATE TEST SET C2='PPPP' WHERE C1= 10;

DBMS\_OUTPUT.PUT\_LINE(SQL%ROWCOUNT);

IF SQL%FOUND THEN

DBMS\_OUTPUT.PUT\_LINE('ENCONTRADO');

END IF;

IF SQL%NOTFOUND THEN

DBMS\_OUTPUT.PUT\_LINE('NO ENCONTRADO');

END IF;

END;

DECLARE

COL1 test.c1%TYPE;

BEGIN

COL1:=10;

INSERT INTO TEST (C1,C2) VALUES (COL1,'AAAA');

COMMIT;

END;

DECLARE

T TEST.C1%TYPE;

BEGIN

T:=10;

UPDATE TEST SET C2='CCCC' WHERE C1=T;

END;

DECLARE

T test.c1%TYPE;

BEGIN

T:=10;

DELETE FROM TEST WHERE C1=T;

COMMIT;

END;

BEGIN

INSERT INTO departments (DEPARTMENT\_ID,DEPARTMENT\_NAME,MANAGER\_ID,LOCATION\_ID) VALUES (300,'ARTE',107,1500);

END;

BEGIN

UPDATE departments SET location\_id=1700 WHERE department\_id=300;

END;

BEGIN

DELETE FROM departments WHERE department\_id=290;

COMMIT;

END;

-- cursor con loop simple

DECLARE

CURSOR C1 IS SELECT \* FROM REGIONS;

V1 REGIONS%ROWTYPE;

BEGIN

OPEN C1;

LOOP

FETCH C1 INTO V1;

EXIT WHEN C1%NOTFOUND;

DBMS\_OUTPUT.PUT\_LINE(V1.REGION\_ID||' '|| V1.REGION\_NAME);

END LOOP;

CLOSE C1;

---------------------------------------------------------

--BUCLE FOR, CUANDO SE USA CON CURSORES ES MAS SENCILLO

FOR i IN C1 LOOP

DBMS\_OUTPUT.PUT\_LINE(i.REGION\_NAME);

END LOOP;

END;

BEGIN

FOR i IN (SELECT \* FROM REGIONS) LOOP

DBMS\_OUTPUT.PUT\_LINE(i.REGION\_NAME||' '||i.region\_id);

END LOOP;

END;

DECLARE

CURSOR C1 (SAL number) IS SELECT \* FROM EMPLOYEES

WHERE SALARY > SAL;

emp1 EMPLOYEES%ROWTYPE;

BEGIN

OPEN C1(10000);

LOOP

FETCH C1 INTO emp1;

EXIT WHEN C1%NOTFOUND;

DBMS\_OUTPUT.PUT\_LINE(emp1.first\_name||' '||emp1.salary);

END LOOP;

DBMS\_OUTPUT.PUT\_LINE('HE ENCONTRADO '||c1%rowcount||' empleados');

CLOSE C1;

END;

-- ACTUALIZAR DATOS CON CURSOR

-- where current of significa que afecte a la fila en la que este en ese momento del cursor

-- rowtype es que convierta la variable del tipo de toda la fila de la tabla asignada

--for update es para poder posteriormente actualizar datos

DECLARE

empl employees%rowtype;

CURSOR cur IS SELECT \* FROM EMPLOYEES FOR UPDATE;

BEGIN

OPEN cur;

LOOP

FETCH cur INTO empl;

EXIT WHEN cur%notfound;

IF empl.COMMISSION\_PCT IS NOT NULL THEN

UPDATE EMPLOYEES SET SALARY=SALARY\*1.10 WHERE CURRENT OF cur;

ELSE

UPDATE EMPLOYEES SET SALARY=SALARY\*1.15 WHERE CURRENT OF cur;

END IF;

END LOOP;

CLOSE cur;

END;

/\*Hacer un programa que tenga un cursor que vaya visualizando los salarios de los empleados.

Si en el cursor aparece el jefe (Steven King) se debe generar un RAISE\_APPLICATION\_ERROR indicando que el sueldo del jefe no se puede ver

\*/

DECLARE

CURSOR C1

IS SELECT first\_name,last\_name,salary from EMPLOYEES;

BEGIN

for i IN C1

LOOP

IF i.first\_name='Steven' AND i.last\_name='King'

THEN

raise\_application\_error(-20300,'El salario del jefe no puede ser visto');

ELSE

DBMS\_OUTPUT.PUT\_LINE(i.first\_name ||' ' || i.last\_name || ': '|| i.salary || 'DLS');

END IF;

END LOOP;

END;

/\*Hacemos un bloque con dos cursores. (Esto se puede hacer fácilmente con una sola SELECT pero vamos a hacerlo de esta manera para probar parámetros en cursores)

o El primero de empleados

o El segundo de departamentos que tenga como parámetro el MANAGER\_ID

o Por cada fila del primero, abrimos el segundo curso pasando el ID del MANAGER

o Debemos pintar el Nombre del departamento y el nombre del MANAGER\_ID

o Si el empleado no es MANAGER de ningún departamento debemos poner “No es jefe de nada”

\*/

SET SERVEROUTPUT ON

DECLARE

DEPARTAMENTO DEPARTMENTS%ROWTYPE;

jefe DEPARTMENTS.MANAGER\_ID%TYPE;

CURSOR C1 IS SELECT \* FROM EMployees;

CURSOR C2(j DEPARTMENTS.MANAGER\_ID%TYPE)

IS SELECT \* FROM DEPARTMENTS WHERE MANAGER\_ID=j;

begin

for EMPLEADO in c1 loop

open c2(EMPLEADO.employee\_id) ;

FETCH C2 into departamento;

if c2%NOTFOUND then

DBMS\_OUTPUT.PUT\_LINE(EMPLEADO.FIRST\_NAME ||' No es JEFE de NADA');

ELSE

DBMS\_OUTPUT.PUT\_LINE(EMPLEADO.FIRST\_NAME || 'ES JEFE DEL DEPARTAMENTO '|| DEPARTAMENTO.DEPARTMENT\_NAME);

END IF;

CLOSE C2;

END LOOP;

END;

/\*Crear un cursor con parámetros que pasando el número de departamento visualice el número de empleados de ese departamento\*/

SET SERVEROUTPUT ON

DECLARE

CODIGO DEPARTMENTS.DEPARTMENT\_ID%TYPE;

CURSOR C1(COD DEPARTMENTS.DEPARTMENT\_ID%TYPE ) IS SELECT COUNT(\*) FROM employeeS

WHERE DEPARTMENT\_ID=COD;

NUM\_EMPLE NUMBER;

BEGIN

CODIGO:=10;

OPEN C1(CODIGO);

FETCH C1 INTO NUM\_EMPLE;

DBMS\_OUTPUT.PUT\_LINE('numero de empleados de ' ||codigo||' es '||num\_emple);

end;

/\*Crear un bucle FOR donde declaramos una subconsulta que nos devuelva el nombre de los empleados que sean ST\_CLERCK. Es decir, no declaramos el cursor sino que lo indicamos directamente en el FOR.\*/

BEGIN

FOR EMPLE IN(SELECT \* FROM EMPLOYEES WHERE JOB\_ID='ST\_CLERK') LOOP

DBMS\_OUTPUT.PUT\_LINE(EMPLE.FIRST\_NAME);

END LOOP;

END;

/\*Creamos un bloque que tenga un cursor para empleados. Debemos crearlo con FOR UPDATE.

o Por cada fila recuperada, si el salario es mayor de 8000 incrementamos el salario un 2%

o Si es menor de 800 lo hacemos en un 3%

o Debemos modificarlo con la cláusula CURRENT OF

o Comprobar que los salarios se han modificado correctamente.

\*/

SET SERVEROUTPUT ON

DECLARE

CURSOR C1 IS SELECT \* FROM EMployees for update;

begin

for EMPLEADO IN C1 LOOP

IF EMPLEADO.SALARY > 8000 THEN

UPDATE EMPLOYEES SET SALARY=SALARY\*1.02

WHERE CURRENT OF C1;

ELSE

UPDATE EMPLOYEES SET SALARY=SALARY\*1.03

WHERE CURRENT OF C1;

END IF;

END LOOP;

COMMIT;

END ;

/

SELECT \* FROM TEST;

EXCEPCIONES

* Ejemplo excepcion

Declare

V\_numero number;

V\_exe exception; -- declarer variable de tipo excepción

Begin

V\_numero := &ingrese\_numero;

If v\_numero > 10 then

Dbms\_output.put\_line(‘el numero es mayor a 10’);

Else

Raise v\_exe; -- si numero es menor a 10 ocurre excepción

End if;

When v\_exe then

Dbms\_output.put\_line(‘el numero no es mayor a 10’);

End;

-- ejemplo2 (pl/sql)

--sql: select first\_name,last\_name from employees where employee\_id=100;

--pl

Declare

v\_nombre varchar2(45);

v\_apellido varchar2(45);

begin

select first\_name,last\_name

INTO v\_nombre,v\_apellido

from employees where employee\_id=&ingrese\_codigo;

Dbms\_output.put\_line('nombre completo: ' || v\_nombre||' '||v\_apellido);

exception

when others then

DBMS\_OUTPUT.PUT\_line('ingrese nuevamente los datos');

end;

-- ejemplo 3) procedimiento almacenado

create or replace procedure sp\_ejemplo

is

-- declaracion de variables

begin

-- codigo de programación

exception

--codigo de error

end;

-- creacion de procedimiento almacenado

create or replace procedure sp\_ejemplo (p\_num1 number,p\_num2 number)

is

v\_total number;

begin

v\_total:= p\_num1+p\_num2;

DBMS\_OUTPUT.PUT\_line('la suma es: ' || v\_total);

end;

execute sp\_ejemplo(2,4);

execute sp\_ejemplo (&ingrese\_num1,&ingrese\_num2);