**Evaluación Final**

Apellidos y Nombres: Anticona Fiestas, Sebastian Miguel

Sección: III.01.2022-IE Sede: Lima Norte

**Pregunta 1: (4pts)**

Ingrese al motor de base de datos, usando los siguientes datos:

|  |  |
| --- | --- |
| **OBJETO** | **DESCRIPCION** |
| TABLESPACE | **FINAL\_ABD**, 150 MB, C:\IDAT\ |
| USUARIO & PASSWORD | USER14 – “123456” |
| ROL | ROL\_USER14 con sus privilegios de acceso y creación de objetos. |

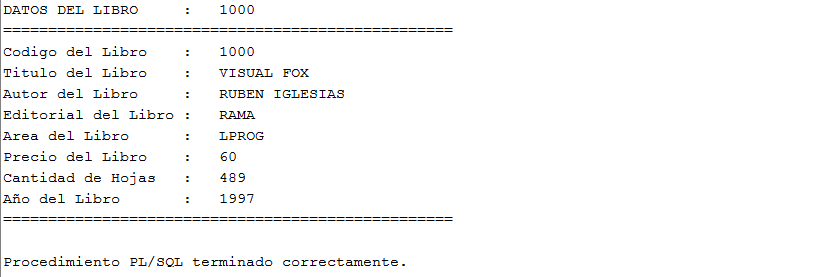
**Facilite el código de dicha cuenta.**

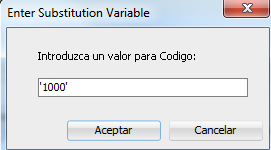
**DESCARGAR EL SCRIPT DE BASE DE DATOS: S14.sql y proceda a ejecutarlo, dado que este tiene toda la estructura de la base de datos.**

**Pregunta 2: (4pts)**

Muestre los datos de un Libro determinado, ingresando mediante una variable de sustitución **&** el código del libro, por ejemplo ‘**1000’**, los datos de dicho libro deben ser mostrados.

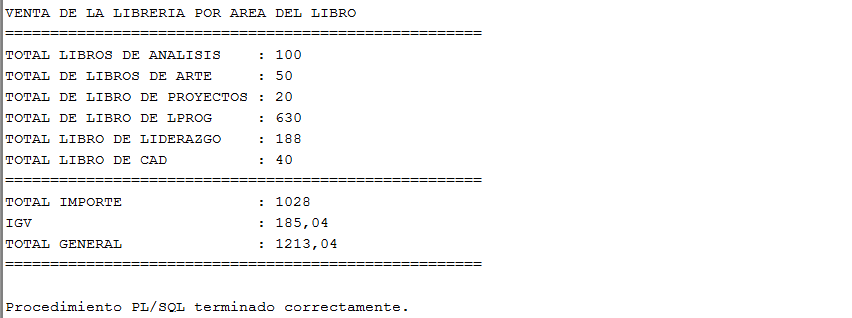
**Si ejecuta este código en Oracle obtendrá el siguiente resultado:**





**Pregunta 3: (4pts)**

Se desea vender todos los libros de la librería, clasificados por área del libro **(ANALISIS, ARTE, PROYECTOS, LPROG, LIDEREZGO y CAD)**, genere un código PL/SQL que realice dicha acción, con totalizaciones incluidas. Redacte el código desde el editor de Oracle PL/SQL.

**Si ejecuta este código en Oracle obtendrá el siguiente resultado:**

**Pregunta 4: (4pts)**

CREAR UN PAQUETE ALMACENADO, EL CUAL CONTENGA TODOS LOS PROCEDIMIENTOS ALMACENADOS DE MANTENIMIENTO DE LA TABLA **ALUMNO**, (grabar, consultar, eliminar, buscar) ESTOS DEBEN FUNCIONAR DESDE EL PAQUETE.

**Pregunta 5: (4pts)**

Elabore los siguientes disparadores:

* Que permita salvar en otra tabla, los registros insertados en la tabla alumno.
* Que cinco alumnos. Pueden tener un solo delegado.
* Que diez trabajadores, solo pueden tener un jefe.

**DESARROLLO:**

Digitar todos los códigos a partir de aquí.

**Solucion\_01:**

**Creación de usuario:**

--1

Create TableSpace SEMANA14 DataFile 'C:\IDAT\IDAT\_14.DBF' Size 150M;

--2

Select \* From v$tablespace;

--3

Create User USER14 Identified by "123456"

Default TableSpace SEMANA14

Temporary TableSpace TEMP

Quota unlimited on SEMANA14;

--4

Create Role ROL\_USER14;

--5

Grant CONNECT, CREATE TABLE, RESOURCE,

ALTER ANY INDEX, ALTER ANY SEQUENCE, ALTER ANY TABLE,

ALTER ANY TRIGGER, CREATE ANY INDEX, CREATE ANY SEQUENCE,

CREATE ANY SYNONYM, CREATE ANY TABLE, CREATE ANY TRIGGER,

CREATE ANY VIEW, CREATE PROCEDURE, CREATE PUBLIC SYNONYM,

CREATE TRIGGER, CREATE VIEW, DROP ANY INDEX,

DROP ANY SEQUENCE, DROP ANY TABLE, DROP ANY TRIGGER, DROP ANY VIEW,

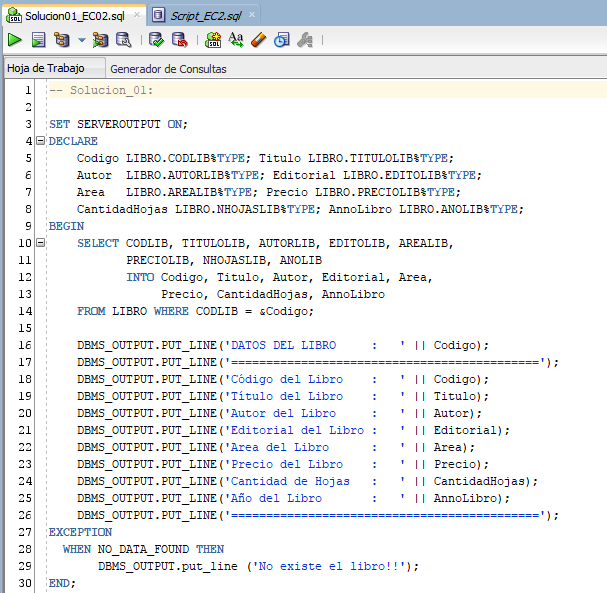
INSERT ANY TABLE, QUERY REWRITE, SELECT ANY TABLE,

UNLIMITED TABLESPACE To ROL\_USER14;

--6

Grant ROL\_USER10X To USER14;

**Solucion\_02:**

****

SET SERVEROUTPUT ON;

DECLARE

Codigo LIBRO.CODLIB%TYPE; Titulo LIBRO.TITULOLIB%TYPE;

Autor LIBRO.AUTORLIB%TYPE; Editorial LIBRO.EDITOLIB%TYPE;

Area LIBRO.AREALIB%TYPE; Precio LIBRO.PRECIOLIB%TYPE;

CantidadHojas LIBRO.NHOJASLIB%TYPE; AnnoLibro LIBRO.ANOLIB%TYPE;

BEGIN

SELECT CODLIB, TITULOLIB, AUTORLIB, EDITOLIB, AREALIB,

PRECIOLIB, NHOJASLIB, ANOLIB

INTO Codigo, Titulo, Autor, Editorial, Area,

Precio, CantidadHojas, AnnoLibro

FROM LIBRO WHERE CODLIB = &Codigo;

DBMS\_OUTPUT.PUT\_LINE('DATOS DEL LIBRO : ' || Codigo);

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

DBMS\_OUTPUT.PUT\_LINE('Código del Libro : ' || Codigo);

DBMS\_OUTPUT.PUT\_LINE('Título del Libro : ' || Titulo);

DBMS\_OUTPUT.PUT\_LINE('Autor del Libro : ' || Autor);

DBMS\_OUTPUT.PUT\_LINE('Editorial del Libro : ' || Editorial);

DBMS\_OUTPUT.PUT\_LINE('Area del Libro : ' || Area);

DBMS\_OUTPUT.PUT\_LINE('Precio del Libro : ' || Precio);

DBMS\_OUTPUT.PUT\_LINE('Cantidad de Hojas : ' || CantidadHojas);

DBMS\_OUTPUT.PUT\_LINE('Año del Libro : ' || AnnoLibro);

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

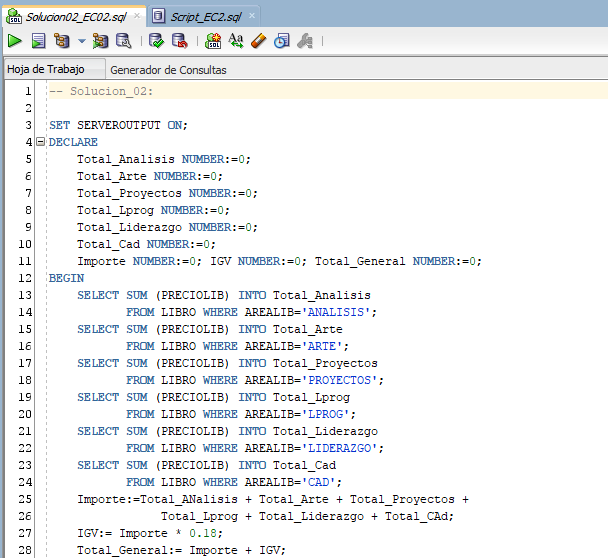
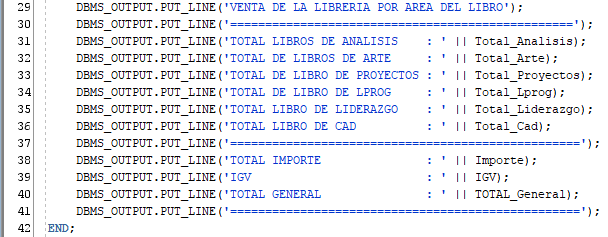
EXCEPTION

WHEN NO\_DATA\_FOUND THEN

DBMS\_OUTPUT.put\_line ('No existe el libro!!');

END;

**Solucion\_03:**



SET SERVEROUTPUT ON;

DECLARE

Total\_Analisis NUMBER:=0;

Total\_Arte NUMBER:=0;

Total\_Proyectos NUMBER:=0;

Total\_Lprog NUMBER:=0;

Total\_Liderazgo NUMBER:=0;

Total\_Cad NUMBER:=0;

Importe NUMBER:=0; IGV NUMBER:=0; Total\_General NUMBER:=0;

BEGIN

SELECT SUM (PRECIOLIB) INTO Total\_Analisis

FROM LIBRO WHERE AREALIB='ANALISIS';

SELECT SUM (PRECIOLIB) INTO Total\_Arte

FROM LIBRO WHERE AREALIB='ARTE';

SELECT SUM (PRECIOLIB) INTO Total\_Proyectos

FROM LIBRO WHERE AREALIB='PROYECTOS';

SELECT SUM (PRECIOLIB) INTO Total\_Lprog

FROM LIBRO WHERE AREALIB='LPROG';

SELECT SUM (PRECIOLIB) INTO Total\_Liderazgo

FROM LIBRO WHERE AREALIB='LIDERAZGO';

SELECT SUM (PRECIOLIB) INTO Total\_Cad

FROM LIBRO WHERE AREALIB='CAD';

Importe:=Total\_ANalisis + Total\_Arte + Total\_Proyectos +

Total\_Lprog + Total\_Liderazgo + Total\_CAd;

IGV:= Importe \* 0.18;

Total\_General:= Importe + IGV;

DBMS\_OUTPUT.PUT\_LINE('VENTA DE LA LIBRERIA POR AREA DEL LIBRO');

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

DBMS\_OUTPUT.PUT\_LINE('TOTAL LIBROS DE ANALISIS : ' || Total\_Analisis);

DBMS\_OUTPUT.PUT\_LINE('TOTAL DE LIBROS DE ARTE : ' || Total\_Arte);

DBMS\_OUTPUT.PUT\_LINE('TOTAL DE LIBRO DE PROYECTOS : ' || Total\_Proyectos);

DBMS\_OUTPUT.PUT\_LINE('TOTAL DE LIBRO DE LPROG : ' || Total\_Lprog);

DBMS\_OUTPUT.PUT\_LINE('TOTAL LIBRO DE LIDERAZGO : ' || Total\_Liderazgo);

DBMS\_OUTPUT.PUT\_LINE('TOTAL LIBRO DE CAD : ' || Total\_Cad);

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

DBMS\_OUTPUT.PUT\_LINE('TOTAL IMPORTE : ' || Importe);

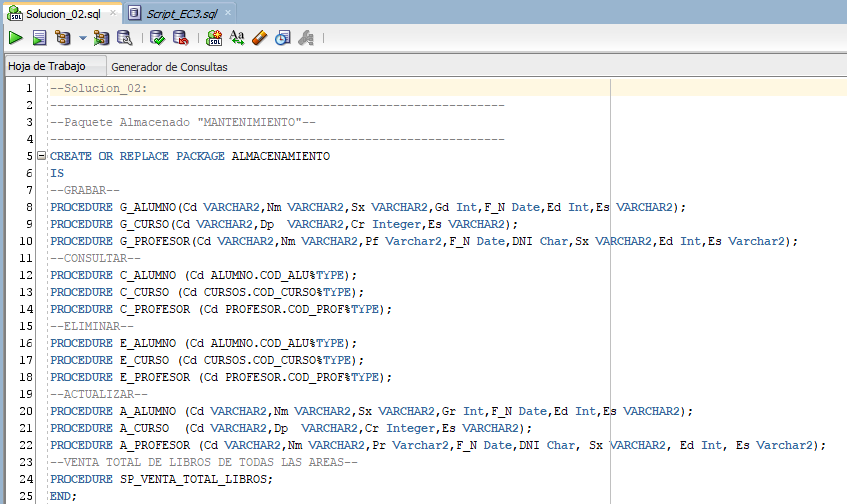
DBMS\_OUTPUT.PUT\_LINE('IGV : ' || IGV);

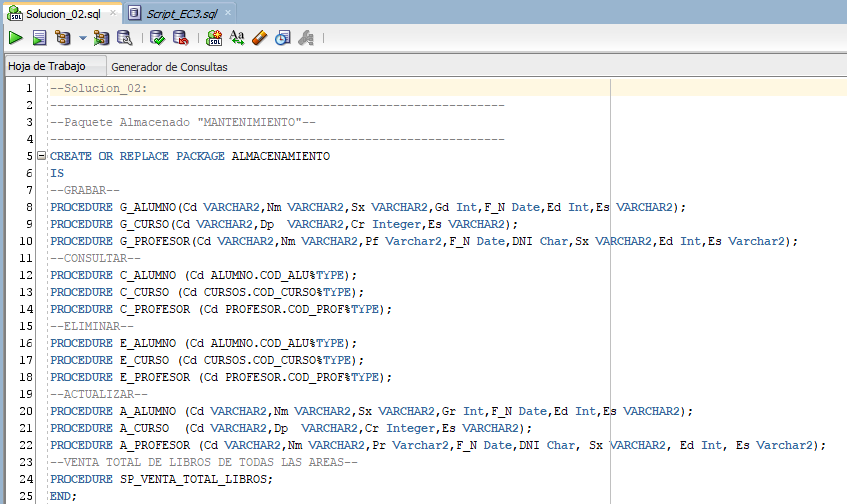
DBMS\_OUTPUT.PUT\_LINE('TOTAL GENERAL : ' || TOTAL\_General);

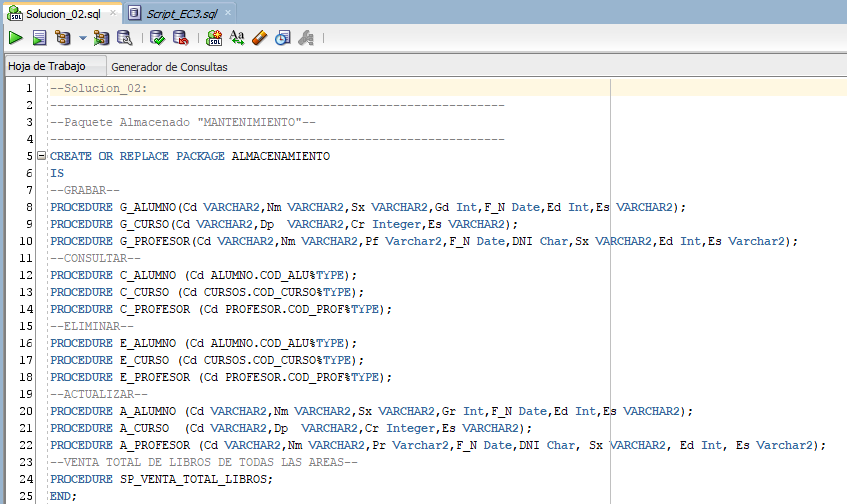
DBMS\_OUTPUT.PUT\_LINE('==================================================');

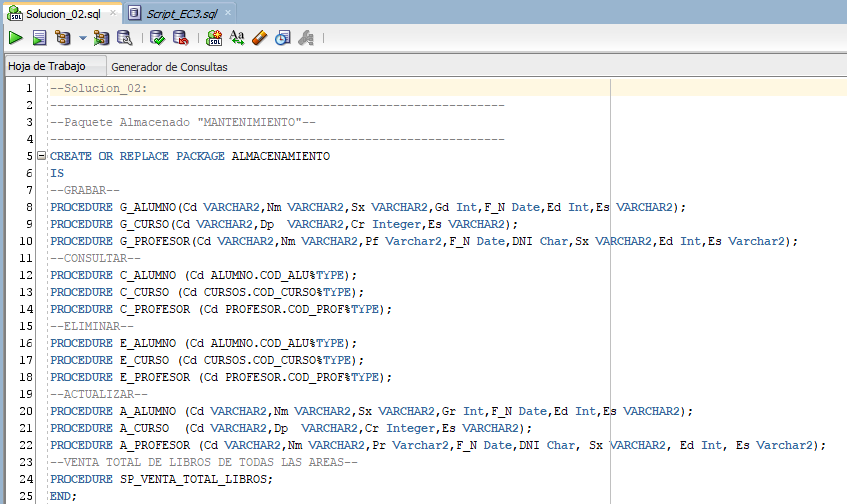
END;

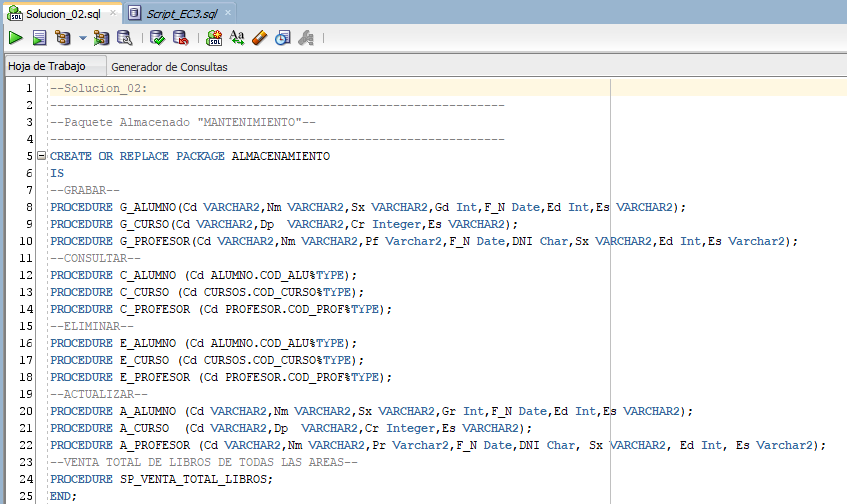
**Solucion\_04:**

****

****

****

****

****

CREATE OR REPLACE PACKAGE ALMACENAMIENTO

IS

--GRABAR--

PROCEDURE G\_ALUMNO(Cd VARCHAR2,Nm VARCHAR2,Sx VARCHAR2,Gd Int,F\_N Date,Ed Int,Es VARCHAR2);

--CONSULTAR--

PROCEDURE C\_ALUMNO (Cd ALUMNO.COD\_ALU%TYPE);

--ELIMINAR--

PROCEDURE E\_ALUMNO (Cd ALUMNO.COD\_ALU%TYPE);

--ACTUALIZAR--

PROCEDURE A\_ALUMNO (Cd VARCHAR2,Nm VARCHAR2,Sx VARCHAR2,Gr Int,F\_N Date,Ed Int,Es VARCHAR2);

END;

CREATE OR REPLACE PACKAGE BODY ALMACENAMIENTO

IS

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

--GRABAR--

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

PROCEDURE G\_ALUMNO(Cd VARCHAR2,

Nm VARCHAR2,

Sx VARCHAR2,

Gd Int,

F\_N Date,

Ed Int,

Es VARCHAR2

)

IS

BEGIN

INSERT INTO ALUMNO(Cod\_Alu,Nombre,Sexo,Grado,F\_Nacimiento,Edad,Estado)Values(Cd,Nm,Sx,Gd,F\_N,Ed,Es);

END G\_ALUMNO;

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

--CONSULTAR--

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

PROCEDURE C\_ALUMNO

( Cd ALUMNO.COD\_ALU%TYPE)

IS

Nm ALUMNO.NOMBRE%TYPE;

Sx ALUMNO.SEXO%TYPE;

Gr ALUMNO.GRADO%TYPE;

F\_N ALUMNO.F\_NACIMIENTO%TYPE;

Ed ALUMNO.EDAD%TYPE;

Es ALUMNO.ESTADO%TYPE;

BEGIN

SELECT NOMBRE INTO Nm FROM ALUMNO WHERE COD\_ALU = Cd;

SELECT SEXO INTO Sx FROM ALUMNO WHERE COD\_ALU = Cd;

SELECT GRADO INTO Gr FROM ALUMNO WHERE COD\_ALU = Cd;

SELECT F\_NACIMIENTO INTO F\_N FROM ALUMNO WHERE COD\_ALU = Cd;

SELECT EDAD INTO Ed FROM ALUMNO WHERE COD\_ALU = Cd;

SELECT ESTADO INTO Es FROM ALUMNO WHERE COD\_ALU = Cd;

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

DBMS\_OUTPUT.PUT\_LINE('============DATOS DEL ALUMNO============');

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

DBMS\_OUTPUT.PUT\_LINE('Codigo : ' || Cd);

DBMS\_OUTPUT.PUT\_LINE('Nombre : ' || Nm);

DBMS\_OUTPUT.PUT\_LINE('Sexo : ' || Sx);

DBMS\_OUTPUT.PUT\_LINE('Grado : ' || Gr);

DBMS\_OUTPUT.PUT\_LINE('F\_Nacimiento : ' || F\_N);

DBMS\_OUTPUT.PUT\_LINE('Edad : ' || Ed);

DBMS\_OUTPUT.PUT\_LINE('Estado : ' || Es);

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

END C\_ALUMNO;

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

--ELIMINAR--

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

PROCEDURE E\_ALUMNO

( Cd ALUMNO.COD\_ALU%TYPE)

IS

BEGIN

DELETE FROM ALUMNO WHERE COD\_ALU = Cd;

COMMIT;

DBMS\_OUTPUT.PUT\_LINE('EL ALUMNO SE ELIMINO DE LA BASE DE DATOS');

END E\_ALUMNO;

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

--ACTUALIZAR--

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

PROCEDURE A\_ALUMNO(Cd VARCHAR2,Nm VARCHAR2,Sx VARCHAR2,Gr Int,F\_N Date,Ed Int,Es VARCHAR2)

IS

BEGIN

UPDATE ALUMNO SET NOMBRE = Nm,

SEXO = Sx,

GRADO = Gr,

F\_NACIMIENTO =F\_N,

EDAD = Ed,

ESTADO = Es

WHERE COD\_ALU = Cd;

COMMIT;

DBMS\_OUTPUT.PUT\_LINE('ALUMNO ACTUALIZADO CORRECTAMENTE');

END A\_ALUMNO;

END ALMACENAMIENTO;

SET SERVEROUTPUT ON;

EXEC ALMACENAMIENTO.G\_ALUMNO('A0015', 'ELVIS ROLANDO FIESTAS','Masculino', 1,'12/02/2002', 20, 'ACTIVO');

EXEC ALMACENAMIENTO.C\_ALUMNO('A0007');

EXEC ALMACENAMIENTO.E\_ALUMNO('A0015');

EXEC ALMACENAMIENTO.A\_ALUMNO('A0003', 'JOSE LUIS RODRIGUEZ','Masculino', 3,'12/02/2000', 22, 'ACTIVO');

**Solucion\_05:**

--Creacion de la nueva tabla:

CREATE TABLE ALUMNOS\_2 (

Cod\_Alu VARCHAR2(5),

Nombre VARCHAR2(50),

Sexo VARCHAR2(20),

Grado Int,

F\_Nacimiento Date,

Delegado VARCHAR2(5),

Edad Int,

Estado Varchar2(20)

);

CREATE OR REPLACE TRIGGER TR\_ALUMNOS

BEFORE INSERT ON Alumno

FOR EACH ROW

BEGIN

INSERT INTO ALUMNOS\_2

VALUES (:new.COD\_ALU, :new.NOMBRE, :new.SEXO, :new.GRADO, :new.F\_NACIMIENTO,

:new.DELEGADO, :new.EDAD, :new.ESTADO);

END;

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

CREATE OR REPLACE TRIGGER TR\_DELEGADOS

BEFORE INSERT ON ALUMNO

FOR EACH ROW

DECLARE

Dele INTEGER;

BEGIN

SELECT count(\*) INTO Dele FROM ALUMNO WHERE DELEGADO = :new.DELEGADO;

IF (Dele > 4) THEN

raise\_application\_error(-20600,:new.DELEGADO||' No se puede insertar más de 5 delegados por alumno');

END IF;

END;

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

CREATE OR REPLACE TRIGGER TR\_JEFES

BEFORE INSERT ON TRABAJADOR

FOR EACH ROW

DECLARE

Jefe INTEGER;

BEGIN

SELECT count(\*) INTO Jefe FROM TRABAJADOR WHERE JEFE = :new.JEFE;

IF (Jefe > 9) THEN

raise\_application\_error(-20600,:new.JEFE||' No se puede ingresar más de 10');

END IF;

END;