

# Ingeniería en Informática

# Sistemas de Bases de Datos

# PL/SQL 02

# Introducción

Docente Tatiana Ilabaca W.  
Segundo semestre de 2023



# Objetivos

Aplicar SELECT...INTO...FROM en bloques de programación

Aplicar los tipos %TYPE y %ROWTYPE en la declaración de variables

Crear y aplicar excepciones en bloques de programación



## SELECT...INTO...FROM

---

- Permite asignar a una o más **variables** el o los valores devueltos por un **SELECT**.
- Para ejecutar **SELECT... FROM** se necesita especificar la cláusula **INTO**, de lo contrario generará un error.
- El resultado del **SELECT** **debe ser una sola fila (o ninguna)**.
- Si el **SELECT** no retorna filas (nulo) se carga la excepción **NO\_DATA\_FOUND**.
- Si el **SELECT** retorna más de una fila se carga la excepción **TOO\_MANY\_ROWS**.

## SELECT...INTO...FROM

- SELECT retorna una fila o ninguna

Mostrar la fecha y hora de una infracción en particular.

```
SET SERVEROUTPUT ON

DECLARE
fecha    DATE;
hora     CHAR(5);
BEGIN
SELECT fechaIN, horaIN INTO fecha, hora
FROM INFRACCION WHERE numeroIN=&NRO_INFRACCION;

DBMS_OUTPUT.PUT_LINE('Fecha: '||fecha);
DBMS_OUTPUT.PUT_LINE('Hora: '||hora);

EXCEPTION
WHEN NO_DATA_FOUND THEN DBMS_OUTPUT.PUT_LINE('Infracción no encontrada');
END;
```

## SELECT...INTO...FROM

- SELECT retorna más de una fila.

Mostrar el rut y género de las personas de nacionalidad chilena.

```
SET SERVEROUTPUT ON

DECLARE
rut      CHAR(9);
genero   CHAR(1);
BEGIN
SELECT rutPR, generoPR INTO rut, genero
FROM PERSONA WHERE codigoNC=1;

EXCEPTION
WHEN TOO_MANY_ROWS THEN DBMS_OUTPUT.PUT_LINE('La consulta devuelve más de una fila');
END;
```

## Tipos %TYPE - %ROWTYPE

- Flexibiliza la definición de los tipos de datos de las variables.
- %TYPE define el tipo de dato de una variable en base al tipo de dato de una **columna** en particular.

HORAIN	CHAR(5 BYTE)
--------	--------------

- %ROWTYPE define el tipo de dato de una variable en base a la **estructura de fila** de una tabla.


NUMEROIN	NUMBER(7,0)
RUTPR	CHAR(9 BYTE)
FECHAIN	DATE
HORAIN	CHAR(5 BYTE)
TIPOIN	VARCHAR2(100 BYTE)
PERMISOTEMP	NUMBER(1,0)
NUMEROPM	NUMBER(8,0)
COVIDDIAG	NUMBER(1,0)
COVIDSOSP	NUMBER(1,0)
COVIDNOTIF	NUMBER(1,0)

## Tipos %TYPE - %ROWTYPE

- **Ejemplo 1:** Mostrar el tipo de infracción para una infracción en particular.

```
DECLARE
tipo    VARCHAR2(100);
BEGIN
SELECT tipoIN INTO tipo FROM INFRACCION WHERE numeroIN=&NRO_INFRACCION;

DBMS_OUTPUT.PUT_LINE('Tipo de infracción: '||tipo);
END;
```



```
DECLARE
tipo    INFRACCION.tipoIN%TYPE;
BEGIN
SELECT tipoIN INTO tipo FROM INFRACCION WHERE numeroIN=&NRO_INFRACCION;

DBMS_OUTPUT.PUT_LINE('Tipo de infracción: '||tipo);
END;
```

## Tipos %TYPE - %ROWTYPE

- **Ejemplo 2:** Mostrar los datos de la comuna de San Antonio.

```
DECLARE
codigo      NUMBER(5,0);
region      NUMBER(2,0);
etapa       CHAR(9);
BEGIN
SELECT codigoCM, codigoRG, etapaCM INTO codigo, region, etapa
FROM COMUNA WHERE nombreCM='SAN ANTONIO';

DBMS_OUTPUT.PUT_LINE('Código: '||codigo);
DBMS_OUTPUT.PUT_LINE('Código de Región: '||region);
DBMS_OUTPUT.PUT_LINE('Etapa: '||etapa);
END;
```

```
DECLARE
filaCom      COMUNA%ROWTYPE;
BEGIN
SELECT * INTO filaCom
FROM COMUNA WHERE nombreCM='SAN ANTONIO';

DBMS_OUTPUT.PUT_LINE('Código: '||filaCom.codigoCM);
DBMS_OUTPUT.PUT_LINE('Código de Región: '||filaCom.codigoRG);
DBMS_OUTPUT.PUT_LINE('Etapa: '||filaCom.etapaCM);
END;
```

nombreVar.nombreCol



```
DECLARE
-- Declaraciones

BEGIN
-- Ejecución

EXCEPTION
-- Excepción

END;
```

## Excepciones

- Corresponde a una advertencia o condición de error.
- Se controlan dentro de su propia sección.
- Cuando ocurre un error se transfiere el control a las sentencias que se encuentren en la sección **EXCEPTION**.
- Una vez finalizada la ejecución de la excepción, no continúa ejecutándose la sección anterior.
- Permite controlar tipos de errores provocados por las sentencias DML, errores aritméticos, de conversión ,etc..

## Tipos de Excepciones

---

- Predefinidas
  - Proporcionadas por el propio lenguaje
  - Controlan las situaciones de error más comunes
  - No se declaran
  - Son válidas en cualquier bloque
  - Ejemplos: NO\_DATA\_FOUND, ZERO\_DIVIDE, etc.
- Definidas por el usuario
  - Deben ser declaradas en la sección DECLARE
  - Se les asigna el tipo EXCEPTION
  - Se lanzan utilizando la sentencia RAISE
  - Es local al bloque donde fue declarada

# Tipos de Excepciones

- **Ejemplo:** Ingresar una persona mayor de edad que no exista en la base de datos.

```
DECLARE
rut          PERSONA.rutPR%TYPE;
existe       NUMBER(1,0);
fechaNac     PERSONA.fechaNacimPR%TYPE;

ERROR_EXISTE EXCEPTION;
ERROR_MENOR_EDAD EXCEPTION;
BEGIN
rut:='&RUT';

SELECT COUNT(rutPR) INTO existe FROM PERSONA WHERE rutPR=rut;

IF(existe=1) THEN
    RAISE ERROR_EXISTE;
ELSE
    fechaNac:='&FECHA_NACIMIENTO';
    IF(EXTRACT(YEAR FROM SYSDATE)-EXTRACT(YEAR FROM fechaNac) < 18) THEN
        RAISE ERROR_MENOR_EDAD;
    ELSE
        DBMS_OUTPUT.PUT_LINE('Datos ingresados correctamente');
    END IF;
END IF;
EXCEPTION
WHEN ERROR_EXISTE THEN DBMS_OUTPUT.PUT_LINE('Persona ya se encuentra ingresada');
WHEN ERROR_MENOR_EDAD THEN DBMS_OUTPUT.PUT_LINE('La persona es menor de edad');
WHEN OTHERS THEN DBMS_OUTPUT.PUT_LINE('Un error inespecífico');
END;
```

1. Implementa cada uno de los ejemplos de esta diapositiva.
2. A medida que vayas avanzando ejecuta y observa las salidas de cada programa.

## Actividad

---