**Trabaja con procedimientos almacenados de SQL**

bookmark\_border

Un *procedimiento almacenado* es un conjunto de instrucciones que se pueden llamar desde otras consultas o desde otros procedimientos almacenados. Un procedimiento puede tomar argumentos de entrada y mostrar valores como resultados. Debes nombrar y almacenar un procedimiento en un conjunto de datos de BigQuery. Un procedimiento almacenado puede acceder a los datos o modificarlos en varios conjuntos de datos de varios usuarios. También puede contener una [consulta de varias instrucciones](https://cloud.google.com/bigquery/docs/multi-statement-queries?hl=es-419).

Algunos procedimientos almacenados están compilados en BigQuery y no necesitan crearse. Estos se llaman *procedimientos del sistema* y puedes obtener más información sobre ellos en la [referencia de procedimientos del sistema](https://cloud.google.com/bigquery/docs/reference/system-procedures?hl=es-419).

Los procedimientos almacenados admiten *declaraciones de lenguaje de procedimiento*, que te permiten realizar variables como definir variables e implementar el flujo de control. Puedes obtener más información sobre las declaraciones de lenguaje de procedimiento en la [referencia del lenguaje de procedimiento](https://cloud.google.com/bigquery/docs/reference/standard-sql/procedural-language?hl=es-419).

**Crea un procedimiento almacenado**

Creación de funciones y procedimientos almacenados

-- funcion que retorna el nombre completo

CREATE FUNCTION dbo.ObtenerNombreCompleto (@DNI INT)

RETURNS VARCHAR(100)

AS

BEGIN

DECLARE @NombreCompleto VARCHAR(100);

SELECT @NombreCompleto = Apellido\_Nombre

FROM DATOS\_PERSONALES\_PRIMARIOS

WHERE DNI = @DNI;

RETURN @NombreCompleto;

END;

SELECT NombreCompleto = Apellido\_Nombre

FROM DATOS\_PERSONALES\_PRIMARIOS

WHERE DNI = 32012345;

--Función para calcular la antigüedad de un emp

--Esta función calcula la antigüedad en años de un empleado dado su Id\_Legajo.

CREATE FUNCTION dbo.CalcularAntiguedad (@Id\_Legajo INT)

RETURNS INT

AS

BEGIN

DECLARE @Fecha\_Ingreso DATE;

DECLARE @Antiguedad INT;

SELECT @Fecha\_Ingreso = Fecha\_Ingreso

FROM LEGAJO

WHERE Id\_Legajo = @Id\_Legajo;

SET @Antiguedad = DATEDIFF(YEAR, @Fecha\_Ingreso, GETDATE());

RETURN @Antiguedad;

END;

--Este procedimiento permite registrar un ascenso para un empleado, ingresando la fecha de --posesión, fecha de ascenso, decreto y fecha

CREATE PROCEDURE dbo.RegistrarAscenso

@Id\_Legajo INT,

@Fecha\_Posesion DATE,

@Fecha\_Ascenso DATE,

@Decreto\_Ascenso VARCHAR(100),

@Proximo\_Ascenso DATE

AS

BEGIN

INSERT INTO ASCENSOS (Id\_Legajo, Fecha\_Posesion, Fecha\_Ascenso, Decreto\_Ascenso, Proximo\_Ascenso)

VALUES (@Id\_Legajo, @Fecha\_Posesion, @Fecha\_Ascenso, @Decreto\_Ascenso, @Proximo\_Ascenso);

END;

### --Procedimiento para registrar una sanción

### --Este procedimiento permite registrar una sanción para un empleado ingresando fecha, --artículo, motivo, recurso días de arrestos, observacion

CREATE PROCEDURE dbo.RegistrarSancion

@Id\_Legajo INT,

@Fecha DATE,

@Articulo INT,

@Motivo VARCHAR(255),

@Recurso VARCHAR(50),

@Dias\_Arresto INT,

@Observacion VARCHAR(255)

AS

BEGIN

IF @Recurso NOT IN ('Presentado', 'No presentado')

BEGIN

RAISERROR('El recurso debe ser "Presentado" o "No presentado"', 16, 1);

RETURN;

END

INSERT INTO SANCIONES (Id\_Legajo, Fecha, Articulo, Motivo, Recurso, Dias\_Arresto, Observacion)

VALUES (@Id\_Legajo, @Fecha, @Articulo, @Motivo, @Recurso, @Dias\_Arresto, @Observacion);

END;

Esta función calcula el número total de días de licencia que un empleado ha usado en un año específico.

CREATE FUNCTION dbo.CalcularDiasLicencia (@Id\_Legajo INT, @Anio INT)

RETURNS INT

AS

BEGIN

DECLARE @DiasLicencia INT;

SELECT @DiasLicencia = SUM(DATEDIFF(DAY, Fecha\_Inicio, Fecha\_Fin) + 1)

FROM LICENCIAS

WHERE Id\_Legajo = @Id\_Legajo

AND YEAR(Fecha\_Inicio) = @Anio;

RETURN COALESCE(@DiasLicencia, 0);

END;

--Este procedimiento permite actualizar el teléfono, celular y correo electrónico en la tabla CONTACTOS\_EXTRAS.

CREATE PROCEDURE dbo.ActualizarContacto

@Id\_Contacto INT,

@Telefono VARCHAR(15),

@Celular VARCHAR(15),

@Email VARCHAR(100)

AS

BEGIN

UPDATE CONTACTOS\_EXTRAS

SET Telefono = @Telefono,

Celular = @Celular,

Email = @Email

WHERE Id\_Contacto = @Id\_Contacto;

END;