Recordando...

Importancia de Base de Datos

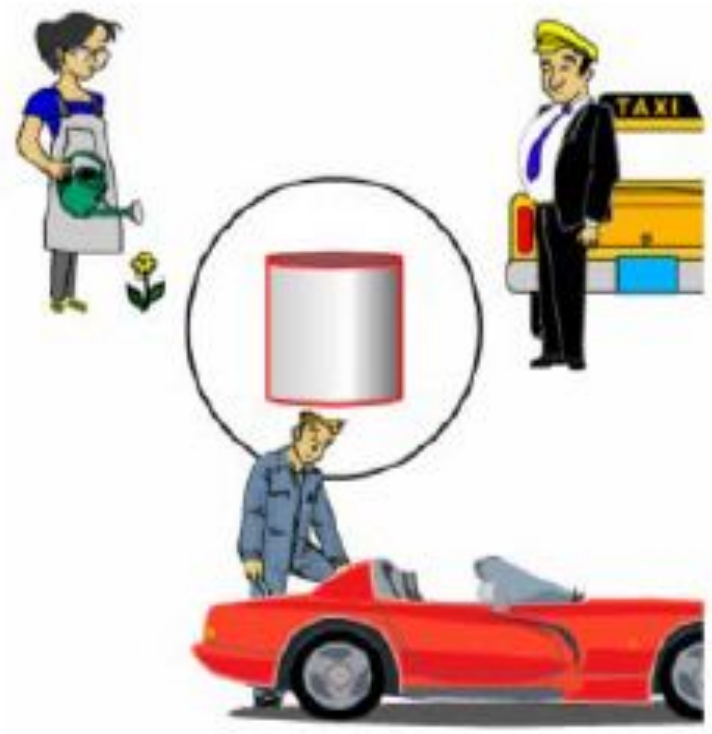
Pregunta: ¿Qué Tiene que Ver una Base de Datos con Mi Vida Diaria?

Recordando...

Importancia de Base de Datos

Si Tiene uno de los Trabajos Siguientes, ¿Cómo Puede Utilizar una Base de Datos?

- Mecánico en un taller de reparación
- Taxista
- Paisajista



Recordando...

Datos, información, base de datos

Recordando...

Datos, información, base de datos

Necesidad de recopilar y mantener datos.

Un sistema de información

Almacena

Procesa datos

automatizar /sistematizar

Objetivo, por medio de una aplicación.
SGBD

Todas las Organizaciones

Base de Datos para Aplicaciones. Objetivo

Optimizar las bases de datos para mejorar el rendimiento de aplicaciones de software.

- **Modelo lógico y físico (Unidad 2-Lección 3 Español Oracle)**
- Índices y vistas
- Consultas avanzadas
- Disparadores
- Procedimientos Almacenados (PA)

Recordando...

Proceso de desarrollo de base de datos

Requisitos de información de negocio

Estrategia y análisis

Modelo de datos conceptuales

Es el examen de un negocio y los datos de negocio con el fin de determinar la estructura de la información de negocio y las reglas que la rigen

Diseño

Diseño de base de datos

Estructura

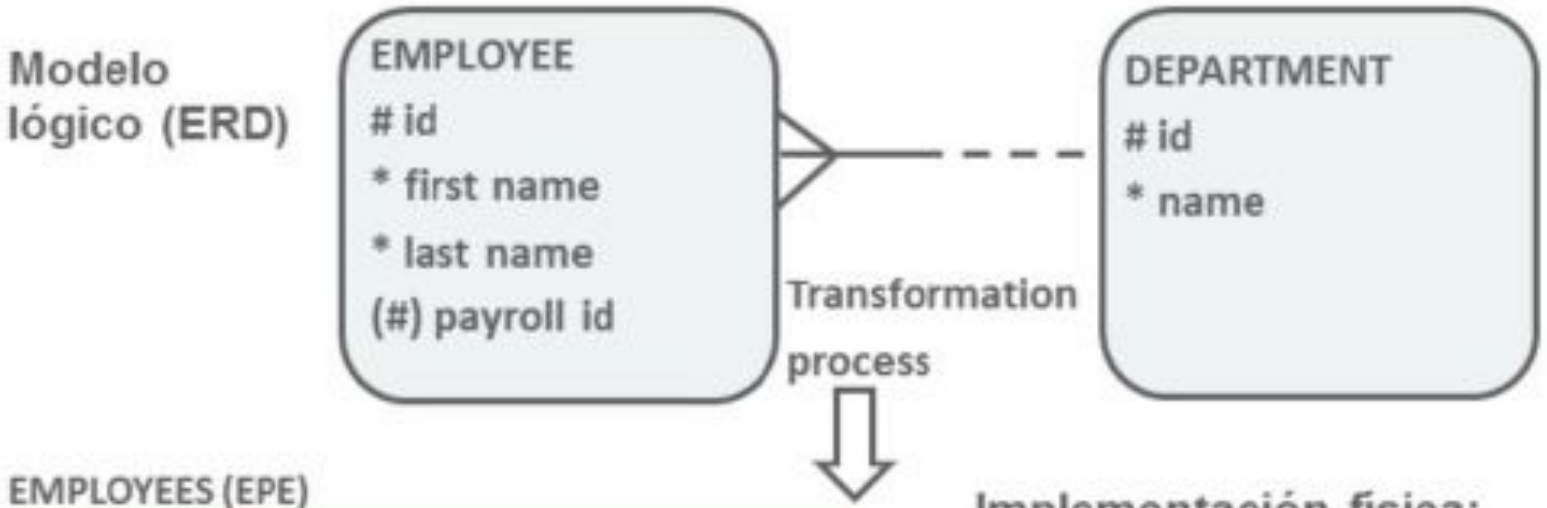
Creación

Creación de base de datos

Modelado de datos físicos, esta relacionado con la implantación de SW y HW, depende de la tecnología y sujeta a cambios la tecnología.

Recordando...

Transformación de lógico a físico. Ejemplo 1



EMPLOYEES (EPE)

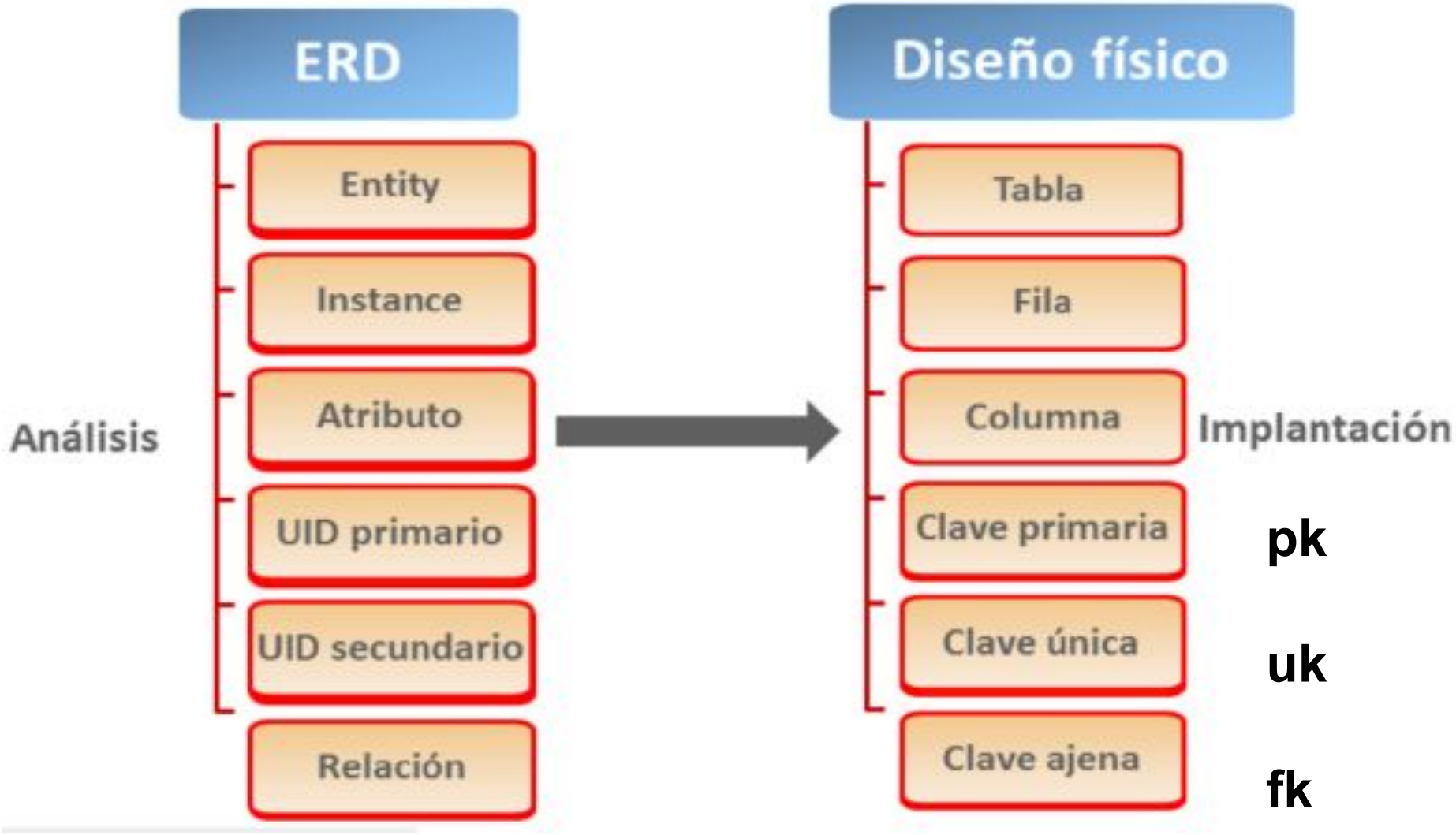
Key Type	Optionality	Column name
pk	*	id
uk	*	payroll_id
	*	last_name
	*	first_name
fk	*	department_id

Implementación física:
Base de datos relacional

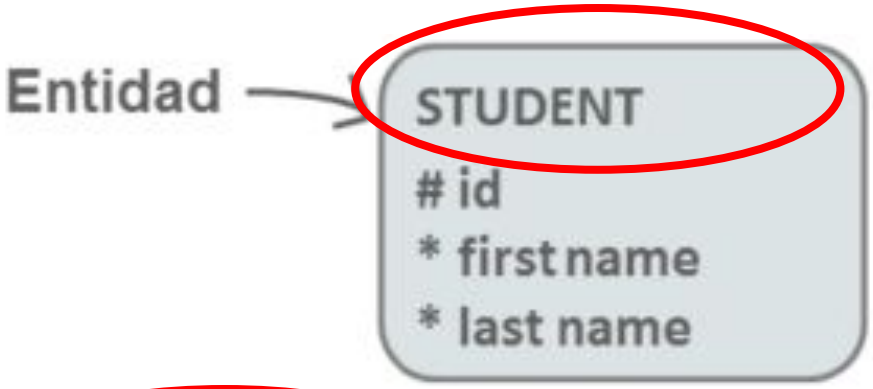
DEPARTMENTS (DPT)

Key Type	Optionality	Column name
pk	*	id
	*	name

Asignación de terminología



Entidad y Tabla (Reglas)



Tabla

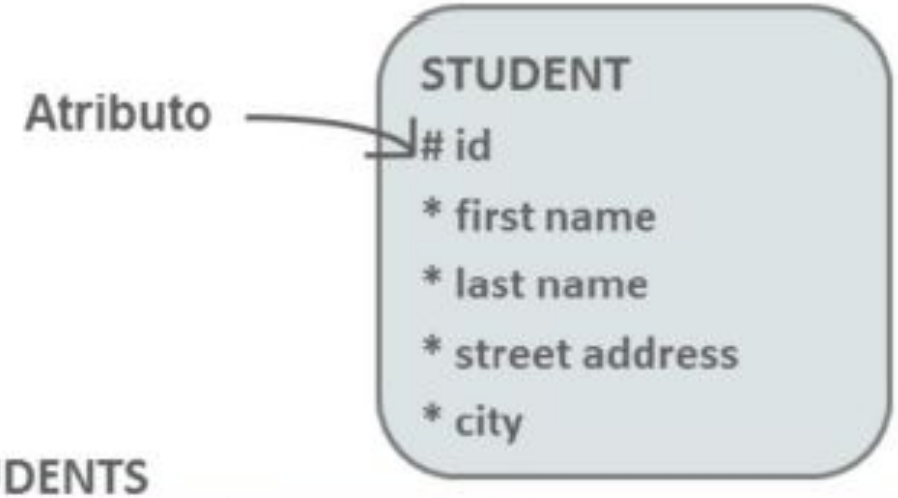
Tabla STUDENTS

ID	FIRST_NAME	LAST_NAME

Antes Singular, **se convierte en plural**

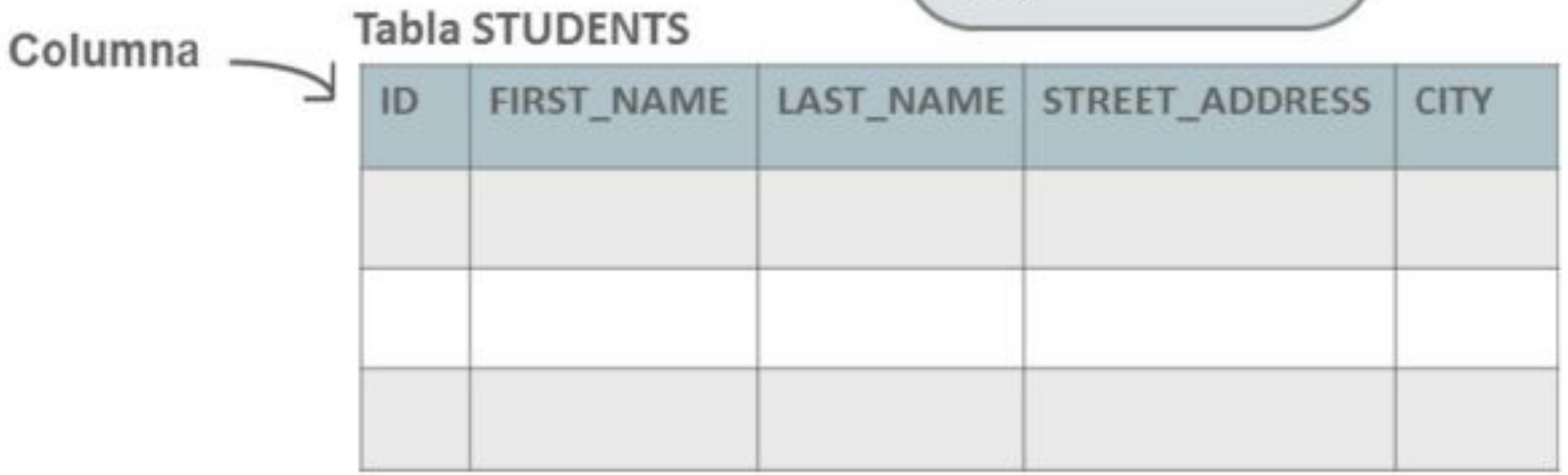
Atributos y columnas (Reglas)

- Nombres de columnas son idénticos, excepto en que los carateres especiales y espacios se sustituyen por "_"



- Nombres de columnas; suelen utilizar más abreviaturas que los nombres de atributos.

First name; First_name o fname



Una instancia y una fila

Entidad	Instance
STUDENT	J Smith

ID	FIRST_NAME	LAST_NAME	STREET_ADDRESS	CITY
101	Sam	Linkin	99B, Chuah Street	LA
102	Neena	Markin	44A, Church Street	NZ
103	Rick	Austina	1st Cross, Palm Street	SA
104	J	Smith	Alpha Street	CA → Fila

Notaciones de diagrama de tabla

Es una documentación adicional; explicar con mayor detalle las claves y columnas de la base de datos física.

Tabla **STUDENTS**

Key Type	Optionality	Column Name
pk	*	id
	*	first_name
	*	last_name
	*	street_address
	*	city

La columna Tipo de clave debe contener valores de **"pk" para la clave primaria, "uk" para la clave única o**

"fk" para la columna de clave ajena.

Base de Datos

¿Qué es SQL?

Lenguaje de Consulta Estructurado

DDL (Lenguaje de Definición de Datos). Define estructuras de B.D.

DML (Lenguaje de Manipulación de Datos). Manipula los datos (INSERT, UPDATE, DELETE)

DQL (Lenguaje de Consulta de Datos). Selecciona datos (SELECT)

DCL (Lenguaje de Control de Datos). Controla el acceso de usuarios

TCL (Lenguaje de Control transaccional). Gestiona las transacciones de B.D.

Caso 1

Una base de datos para una pequeña empresa debe contener información acerca de clientes, artículos y pedidos. Hasta el momento se registran los siguientes datos en varios documentos, para cada cliente se necesita un número de cliente (único), direcciones de envío (varias por cliente), saldo, límite de crédito (depende del cliente, pero en ningún caso debe superar los 3,000 pesos) y descuento en caso de tenerlo.

Caso 1

Para cada artículo se necesita un número de artículo (único) las fábricas que lo distribuyen, existencias de ese artículo en cada fábrica y descripción del artículo, para cada pedido de necesita tener una cabecera y el cuerpo del pedido, la cabecera está formada por el número de cliente, dirección de envío y fecha del pedido. El cuerpo del pedido son varias líneas, en cada línea se especifican el número del artículo pedido y la cantidad. Además, se ha determinado que se debe almacenar la información de las fábricas.

Caso 1

Sin embargo, dado el uso de distribuidores, se usará el número de la fábrica (único) y teléfono de contacto. Y se desean ver cuántos artículos (en total) provee la fábrica. También, por información estratégica, se podría incluir información de fábricas alternativas respecto de las que ya fabrican artículos para esta empresa.

Nota: Una dirección se entenderá como calle, número, colonia y ciudad.

Caso 2

Una organización no gubernamental se encarga de enviar ayuda material (medicamentos y alimentos) y ayuda humanitaria (personal sanitario) a campos de refugiados. Esta organización obtiene sus ingresos de las cuotas de los socios, de los que se desea conocer los datos personales, la cuenta bancaria en donde se realizan los cargos anuales, la fecha de pago y el tipo de cuota. En la actualidad hay tres tipos de cuotas, pudiendo variar en el futuro: mínima (10 mil pesos anuales), media (20 mil pesos anuales) o máxima (30 mil pesos anuales).

Caso 2

Cada socio pertenece a una de las sedes de la organización, cada una de ellas ubicada en una ciudad distinta. De las sedes se desea conocer el domicilio y el nombre de su director. La organización cuenta con dos tipos de voluntarios: los que realizan labores humanitarias (personal sanitario) y los que realizan labores administrativas (personal administrativo). De los primeros se desea conocer su profesión (médico, cedula, etc.), su disponibilidad actual (sí/no) y el número de trabajos en los que ha participado. De todos los voluntarios se desea conocer los datos personales y la sede en la que se inscribieron.

Caso 2

Cada envío tiene un destino y una fecha de salida. Para identificar los envíos, se les asigna un código único. Además, cada envío es organizado por una o varias sedes. Los envíos de ayuda material pueden ser de alimentos, debiéndose conocer el número de toneladas de cada alimento que se manda; o pueden ser de medicamentos, debiéndose conocer el número de unidades de cada medicamento. De los envíos de ayuda humanitaria se debe conocer el número de voluntarios que se mandan de cada profesión y quienes son cada uno de ellos.

Índices y vistas

Objetivo

- Comprender el concepto de **índice**, **crear y eliminar.**
- Generar **vistas** a través de consultas de base de datos.

ÍNDICES. Concepto

Tiene un funcionamiento similar al índice de un libro puesto que, se guardan en parejas de elementos: el **elemento** que se desea indexar y su **posición** en la base de datos.

ÍNDICES. Concepto

Es una estructura de datos que mejora la **velocidad de las operaciones,** permitiendo un **rápido acceso a los registros** de una tabla. Los índices se suelen usar sobre aquellos campos sobre los cuales se vayan a realizar **búsquedas frecuentes.**

Ventaja	Desventaja
• **Mejoran** el rendimiento de los **SELECT**	• **Empeoran** (mínimamente) el rendimiento de los **INSERT** y los **DELETE.**

ÍNDICES. Concepto

Se recomienda el uso de índices cuando la tabla contiene miles de registros o se realizan operaciones de ordenamiento y agrupamiento, etc.

Es importante identificar el o los campos por los que sería útil crear un índice, aquellos campos por los cuales se realizan búsquedas con frecuencia: **claves primarias, claves externas o campos que combinan tablas**.

No se recomienda crear índices sobre campos que no se usan con frecuencia en consultas o en tablas muy pequeñas.

ÍNDICES. Creados automáticamente

Al crearse una tabla en Oracle se crea:

- Un índice UNIQUE que es para mantener las columnas que se hayan definido como **clave primaria de la tabla** utilizando **CONSTRAINT PRIMARY KEY.**

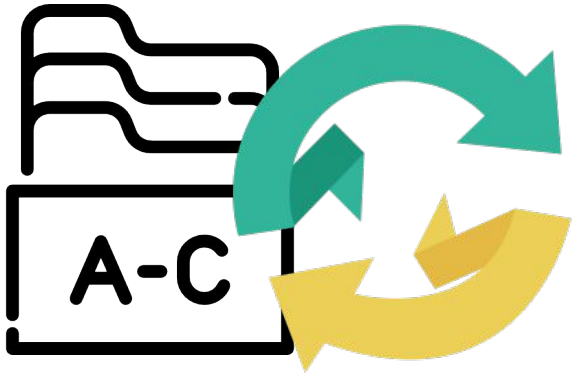
- Un índice UNIQUE es para mantener la **restricción de unicidad** de cada grupo de columnas que se haya declarado como único.

- Un índice para mantener todas las **filas de una tabla organizada por índice**.

ÍNDICES. Consideraciones

- Un índice **sólo es efectivo cuando es utilizado**.

- El mantenimiento de un índice tiene efecto sobre el rendimiento de operaciones de **eliminación, inserción y actualización**. (**Cada que se realiza una de estas operaciones, se actualiza el índice**).



ÍNDICES. Sintaxis

create */unique/* index NOMBREINDICE on NOMBRETABLA(CAMPOS);

unique indica que el valor de la o las columnas indexadas debe ser único, **no puede haber duplicidades**.

nombre_índice debe ser un nombre claro o preciso (no debe existir otro nombre de objeto en Oracle) que siga los convenios de denominación de Oracle para nombrar columnas.

nombre_tabla será el nombre de la tabla donde se creará el índice.

nombre_columna (o columnas) será la columna de la tabla *nombre_tabla* en la que se creará el índice. Se puede crear un índice para varias columnas.

B.D. Empresa

ÍNDICES. Ejercicios

Consultar "user_constraints":

```
select constraint_name, constraint_type, index_name
from user_constraints  where table_name='EMPLEADO';
```

Ver los índices de la tabla "empleados":

```
select index_name, index_type, uniqueness
 from user_indexes
 where table_name='EMPLEADO';
```

Aparece 1 fila, mostrando el nombre del índice, indicando que es normal y único

ÍNDICES. Ejercicios

Creamos un índice único sobre el campo "Apellido1":

```
create unique index I_empleados_Apellido1 on
empleados(Apellido1);
```

Aparecen 2 filas, una por cada índice.

Ver los índices de la tabla "empleado":

```
select index_name, index_type, uniqueness
  from user_indexes
  where table_name='EMPLEADO';
```

B.D. Empresa

ÍNDICES. Ejercicios

Agregamos a la tabla una restricción única sobre el campo "Apellido1":

```
 alter table empleado
   add constraint UQ_empleado_Apellido1
   unique(Apellido1);
--Analicemos la información que nos muestra "user_constraints":

 select constraint_name, constraint_type, index_name
  from user_constraints
  where table_name='EMPLEADO';
```

En la columna "index_name" correspondiente a la restricción única, aparece "I_empleados_apellido1", Oracle usa para esta restricción el índice existente, no crea otro nuevo.

B.D. Empresa

ÍNDICES. Ejercicios

```
--Creamos un índice no único, compuesto (para los campos "apellido"
y "nombre"):

 create index I_empleado_apellido1nombre
 on empleado(apellido1,nombre);

--Consultamos el diccionario "user_indexes":

 select index_name, index_type, uniqueness
  from user_indexes
  where table_name='EMPLEADO';
```

Nos muestra información sobre los 3 índices de la tabla.

B.D. Empresa

ÍNDICES. Ejecución

Para ver el índice activos creados se ejecuta la siguiente consulta:

Sintaxis

```
select index_name Nombre, index_type Tipo,
table_name Tabla, tablespace_name Tablespace,
secondary Secundario
from all_indexes
where table_name = ‘NOMBREDETABLA';
```

B.D. Empresa

ÍNDICES. Ejecución

```
--Veamos todos los índices de la base de datos
activa consultando "user_objects":

 select *from user_objects
  where object_type='INDEX';
```

Aparecen varios índices, entre ellos, los de nuestra tabla "empleados".

B.D. Empresa

ÍNDICES. Ejecución

```
--Obtenemos información de
"user_ind_columns":

 select
index_name,column_name,column_position
  from user_ind_columns
  where table_name='EMPLEADO';
```

index_name	column_name	column_position
PK_EMPLEADO	DNI	1
I_EMPLEADO_APELL	APELLIDO1	1
I_EMPLEADO_APELL	APELLIDO1	1
I_EMPLEADO_APELL	NOMBRE	2

ÍNDICES. Ejecución

```
  --Si intentamos crear un índice único para el campo
"apellido"
  --(que contiene valores duplicados") Oracle no lo permite:

 create unique index I_empleados_apellido1
 on empleado(apellido1);
--error:dicha lista de columnas ya está indexada
--Igualmente, si hay un índice único sobre un campo y luego
--intentamos ingresar un registro con un valor repetido para
el campo indexado, Oracle no lo permite.
```

B.D. Empresa

ÍNDICES. Ejecución

```
--Creamos un índice único sobre el campo "nombre":

 create unique index I_empleado_nombre  on empleado(nombre);
--Oracle lo permite porque no hay valores duplicados.
--Intentamos agregamos un registro que repita un nombre:

insert into empleado values('Araceli','Jacobo','Martínez',01,
to_date('12-16-2022','mm-dd-yyyy'),'zapata','M',3000,888665555,1);

 --Oracle si inserta, pero despues ya no
insert into empleado
values('Araceli','Jacobo','Martínez',02,to_date('12-16-2022','mm-dd-yyyy'),
'zapata','M',3000,888665555,1);      --Oracle no lo permite.
```

B.D. Empresa

ÍNDICES. Monitoreo

```
---Cómo saber si un índice se está utilizando,
monitorización del uso de índices en Oracle
alter index I_EMPLEADO_NOMBRE monitoring usage;

SELECT * FROM V$OBJECT_USAGE;
```

Resultado de la Consulta | Salida de Script | Salida de DBMS | Explicación del plan | Rastreo automático | Historial SQL | Carga de

Descargar ▾ Tiempo de ejecución: 0.003 segundos

index_name	table_name	monitoring	used	start_monitoring	end_monitoring
I_EMPLEADO_APELL	EMPLEADO	YES	NO	05/16/2022 17:22:2	(nulo)
I_EMPLEADO_NOM	EMPLEADO	YES	NO	05/16/2022 17:27:4	(nulo)

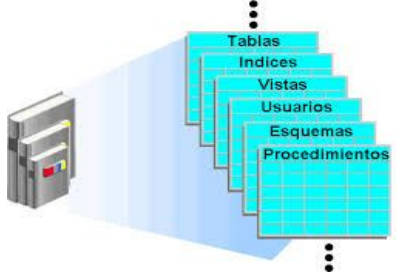
ÍNDICES. Eliminación

Para eliminar un índice se utiliza la siguiente sintaxis:

drop index NOMBREINDICE;

Ejemplo:

drop index `I_empleado_apellido1nombre`



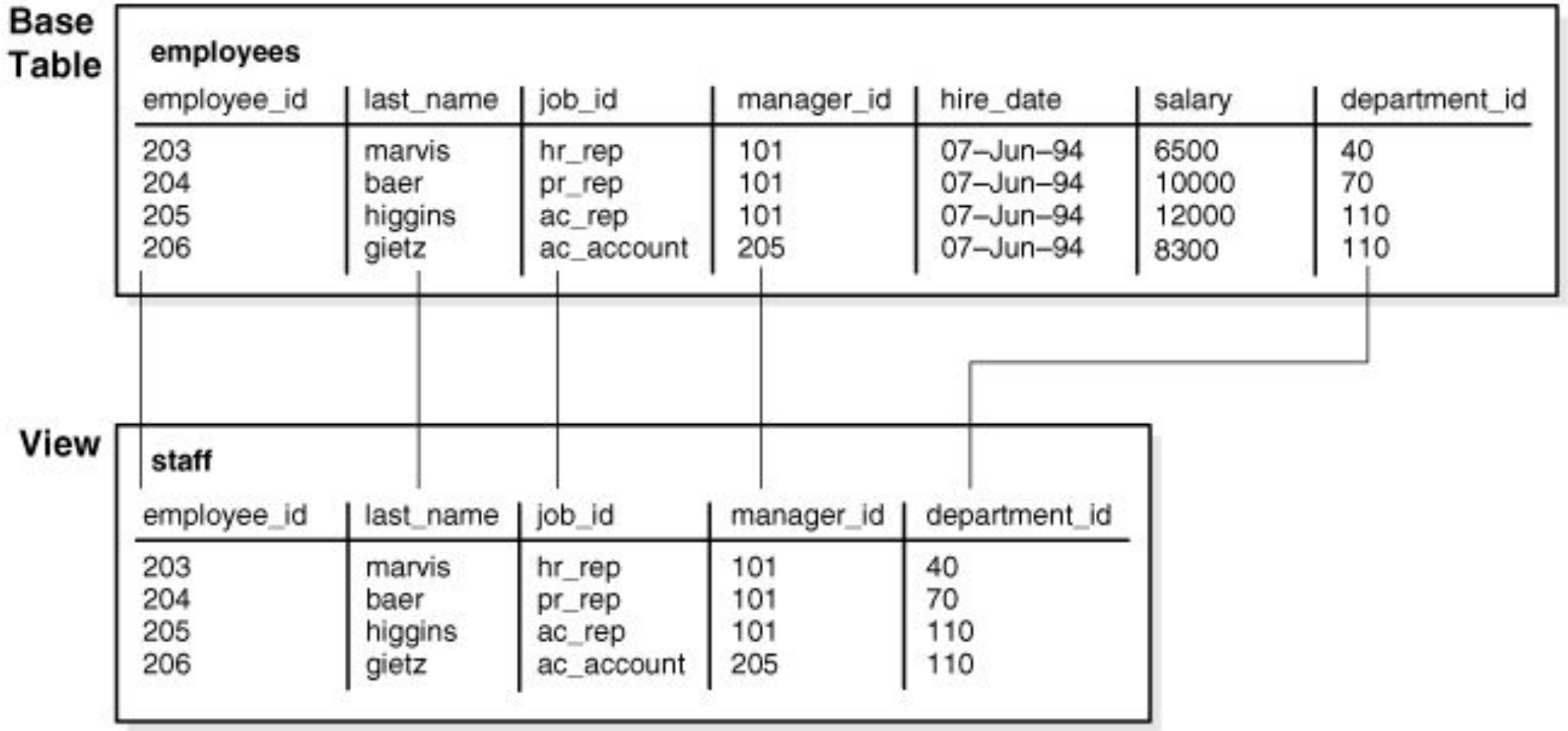
Vistas. Concepto

Una **vista** es una alternativa para mostrar datos de varias tablas; es como una tabla virtual que almacena una consulta.

Los datos accesibles a través de la vista no son almacenados en la base de datos, solo se **guarda** la definición de la vista y no el resultado de ella.

Una vista **almacena** una consulta como un objeto para utilizarse posteriormente.

Vistas. Concepto



Vistas. Sintaxis

```
create view nombrevista as select columna1, columna2, …
    from nombretabla
        where condición;
```

Para mostrar el contenido de una vista se utiliza la siguiente sintaxis:

```
select *from nombrevista;
```

B.D. Empresa

Vistas. Ejercicios

```
--VISTAS


create view VISTAPROYECTO
as select NOMBREPROYECTO,NUMPROYECTO
from PROYECTO
where NUMPROYECTO=1;


-- SINTAXIS PARA OBSERVAR RESULTADO DE VISTA
select *from VISTAPROYECTO;
```

Vistas. Sintaxis para cambiar una vista

create or replace view nombrevista as select columna1, columna2, from nombretabla where condición;

```
create or replace view vistaproyecto

as select NOMBREPROYECTO,NUMPROYECTO  from proyecto

where NUMPROYECTO=10;
```

Vistas. Sintaxis visualizar, eliminar

Sintaxis para visualizar todas las vistas creadas

```
select *from user_objects where object_type='VIEW';
```

Sintaxis para eliminar

```
drop view nombredelavista;
```

```
drop view vistaproyecto;
```

B.D. Empresa

Ejercicios

1. Crear la vista **"vista_EmpleadoProyec",** que sea resultado de una combinación de las dos tablas,en la cual se muestran más de 3 campos.

2. Ver la información de la vista realizada.

3. Realizar una consulta a la vista como si se tratara de una tabla, ordenando de forma ascendente.

4. Realiza una vista donde se pueda visualizar el nombre, apellido y sueldo de cada empleado.

5. De la vista anterior, realizar una consulta donde solo se visualicen los empleados que ganan 25000.

B.D. Empresa

Count

Devuelve el número de filas de la consulta, es decir, el número de registros que cumplen una determinada condición, se puede aplicar a cualquier tipo de dato.

Sintaxis

```
select count (columna) from tabla;
```

```
-- cuantas tuplas tiene la tabla trabaja_en
select count(*) from TRABAJA_EN;
-- cuantas tuplas son las que tienen registro
en horas
select count(HORAS) from TRABAJA_EN;
```

También se puede utilizar esta función junto con la cláusula "where" para una consulta más específica.

```
select count(columna)
  from tabla
  where condiciónwhere;
```

```
select count(*)
from EMPLEADO
where SUELDO='25000';
```

B.D. Empresa

Sum

La función sum retorna la suma de los valores que contiene el campo especificado, se puede emplear solo a datos numéricos.

Sintaxis

```
select sum (columna) from tabla;
```

```
select sum (SUELDO)
from EMPLEADO;
```

También se puede utilizar esta función junto con la clausula "where" para una consulta más específica.

```
select sum (columna)
  from tabla
  where condiciónwhere;
```

B.D. Empresa

Max Min

Para averiguar el valor máximo o mínimo de un campo usamos las funciones max y min respectivamente, se puede emplear con cualquier tipo de dato.

Sintaxis

```
select max o min (columna) from tabla;
```

También se puede utilizar esta función junto con la clausula "where" para una consulta más específica.

```
select max o min(columna)
  from tabla
  where condiciónwhere;
```

B.D. Empresa

AVG

La función avg retorna el valor promedio de los valores del campo especificado, se puede emplear solo a datos numéricos.

Sintaxis

```
select avg (columna) from tabla;
```

También se puede utilizar esta función junto con la clausula "where" para una consulta más específica.

```
select avg (columna)
  from tabla
  where condiciónwhere;
```

B.D. Empresa

GRUPU BY

La cláusula GROUP BY es un comando que se usa para agrupar filas que tienen los mismos valores.

```
select  (editorial) from libros group by editorial;
```

Sintaxis

```
select (columna) from tabla group by columna;
```

La cláusula GROUP BY se utiliza en la instrucción SELECT. Opcionalmente se usa junto con funciones agregadas para producir informes resumidos de la base de datos.

```
select columna, funciondeagregado
  from tabla
  group by columna;
```

```
select editorial, sum(cantidad)
  from libros
  group by editorial;
```

```
select editorial, count(*)
  from libros
  where precio<30
  group by editorial;
```

B.D. Empresa

HAVING

Permite seleccionar o rechazar un grupo de registros.

La cláusula HAVING se usa en combinación con la cláusula GROUP BY para restringir los grupos de filas devueltas solo a aquellos cuya condición es VERDADERA.

```
SELECT nombrecolumna
    agregar_function (SUM, COUNT, MIN, MAX, or AVG) (nombrecolumna)
FROM nombretabla
[WHERE condicion]
GROUP BY nombrecolumna
HAVING  AVG condicion_having;
```

Así como la cláusula "where" permite seleccionar (o rechazar) registros individuales; la cláusula "having" permite seleccionar (o rechazar) un grupo de registros.

Trigger (Disparador)

Un "trigger" (disparador o desencadenador) es un bloque de código que se ejecuta automáticamente cuando ocurre algún evento como **insert**, **update** o **delete** sobre una determinada tabla.

Trigger (Disparador)

- ✔No pueden ser invocados directamente.
- ✔No reciben y retornan parámetros.
- ✔Son apropiados para mantener la integridad de los datos.

Trigger (Disparador)

```
CREATE [ OR REPLACE ] TRIGGER NOMBRETRIGGER

MOMENTO (BEFORE, AFTER o INSTEAD OF ) , EVENTO (INSERT,
UPDATE o DELETE)

  ON NOMBRETABLA

  NIVEL (STATEMENT o FOR EACH ROW ]

WHEN CONDICION--OPCIONAL
```

```
BEGIN

  SENTENCIAS A EJECUTAR

END;

/
```

UNIVERSIDAD TECNOLÓGICA
EMILIANO ZAPATA DEL ESTADO DE MORELOS

Trigger (Disparador)

- Si se agrega "or replace" al momento de crearlo y ya existe un trigger con el mismo nombre, el trigger será borrado y vuelto a crear.

- Momento (indica cuando se disparará el trigger en relación al evento)
  - Before: Significa que el trigger se activará antes que se ejecute la operación (insert, update o delete) sobre la tabla, que causó la activación del mismo.
  - After: significa que el trigger se activará después que se ejecute la operación que causó la activación.
  - Instead of: sólo puede definirse sobre vistas, anula la sentencia disparadora, se ejecuta en lugar de tal sentencia (ni antes ni después).

UNIVERSIDAD TECNOLÓGICA
EMILIANO ZAPATA DEL ESTADO DE MORELOS

Trigger (Disparador)

- Evento: Especifica la operación que causa que el trigger se active, puede ser "insert", "update" o "delete".

- Nombretabla: Indica la tabla asociada al trigger.

Nivel puede ser a nivel de sentencia o de fila.

For each row: Indica que el trigger es a nivel de fila, es decir, se activa una vez por cada registro afectado por la operación sobre la tabla.

Statement: Se activa una sola vez; antes o después de ejecutar la operación sobre la tabla.

UNIVERSIDAD TECNOLÓGICA
EMILIANO ZAPATA DEL ESTADO DE MORELOS

Trigger (Disparador)

- Cuerpo del trigger: Son las acciones que se ejecutan al activarse el trigger, las condiciones que determinan cuando un intento de inserción, actualización o borrado provoca las acciones que el trigger realizará.

UNIVERSIDAD TECNOLÓGICA
EMILIANO ZAPATA DEL ESTADO DE MORELOS

Trigger (Disparador)

- Cuando se trabaja con un trigger a nivel de fila, Oracle provee dos tablas temporales a las cuales se puede acceder.
- Estas tablas contienen los **antiguos** y **nuevos** valores de los campos del registro afectado por la sentencia que activó el trigger.
- Para especificar el campo se utiliza la siguiente sintaxis:

:new.CAMPO :old.CAMPO

UNIVERSIDAD TECNOLÓGICA
EMILIANO ZAPATA DEL ESTADO DE MORELOS

Trigger (Disparador)

Tipo de evento	Campo accedido
Insert	:new
update	:new y :old
delete	:old

UNIVERSIDAD TECNOLÓGICA
EMILIANO ZAPATA DEL ESTADO DE MORELOS

Trigger (Disparador)

- El valor de ":new" puede modificarse en un **trigger before**, es decir, se puede acceder a los nuevos valores antes que se ingresen en la tabla y cambiar los valores asignando a ":new.CAMPO" otro valor.

- El valor de ":new" **NO** puede modificarse en un **trigger after**, esto es porque el trigger se activa luego que los valores de "new" se almacenaron en la tabla.

- El campo ":old" **nunca** se modifica, sólo puede leerse.

UNIVERSIDAD TECNOLÓGICA
EMILIANO ZAPATA DEL ESTADO DE MORELOS

Trigger (Disparador)

- En los triggers a nivel de fila, se puede incluir una restricción adicional, agregando la cláusula **"when"** con una condición que se evalúa para cada fila que afecte el trigger; si resulta **cierta**, **se ejecutan** las sentencias del trigger para ese registro; si resulta **falsa**, el trigger **no se activa** para ese registro.

UNIVERSIDAD TECNOLÓGICA
EMILIANO ZAPATA DEL ESTADO DE MORELOS

Trigger (Disparador)

- Un trigger puede estar en dos estados: habilitado o deshabilitado.
- Sintaxis para deshabilitar un trigger:

```
alter trigger nombretrigger disable;
```

- Sintaxis para habilitar un trigger que está deshabilitado:

```
alter trigger nombretrigger enable;
```

UNIVERSIDAD TECNOLÓGICA
EMILIANO ZAPATA DEL ESTADO DE MORELOS

Trigger (Disparador)

- Para eliminar un trigger se emplea la siguiente sintaxis:

```
drop trigger nombretrigger;
```

- Si se elimina una tabla, se eliminan todos los triggers establecidos sobre ella.

UNIVERSIDAD TECNOLÓGICA
EMILIANO ZAPATA DEL ESTADO DE MORELOS

Procedimiento almacenado

Un procedimiento almacenado es un conjunto de instrucciones a las que se les da un nombre, el cual se almacena en la base de datos activa.

Permitiendo agrupar y organizar tareas repetitivas.

Procedimiento almacenado

Ventajas:

- Incrementa el rendimiento de las aplicaciones.
- Reduce el tráfico entre la aplicación y el servidor de BD.
- Son reutilizables.
- Son seguros.

Desventajas:

- Si se abusa de su uso, incrementa el uso de memoria significativamente.
- Si las reglas de negocio de un procedimiento almacenado son complicadas, son difíciles de implementar.
- Es difícil depurarlos.

Procedimiento almacenado

Generalidades:

- Un procedimiento almacenado se invoca llamándolo.
- Puede hacer referencia a tablas, vistas, a funciones definidas por el usuario o a otros procedimientos almacenados.

Procedimiento almacenado

Sintaxis:

```
CREATE [OR REPLACE]
 PROCEDURE nombreprocedimiento [(
   <parametro1> [IN|OUT|IN OUT] <tipodato>,
   <parametro2> [IN|OUT|IN OUT] <tipodato>)]
 IS / AS
--Declaración de variables locales
-- BEGIN
  -- Sentencias
  [EXCEPTION]
  -- Sentencias control de excepción
END;
```

Procedimiento almacenado

Parámetro: Es el nombre que se le quiere dar al parámetro. Se pueden utilizar múltiples parámetros. En caso de no necesitarlos, se puede omitir los paréntesis.

IN: Especifica que el parámetro es de entrada y que por tanto dicho parámetro tiene que tener un valor en el momento de llamar a la función o procedimiento. Si no se especifica nada, los parámetros son por defecto de tipo entrada.

OUT: Especifica que se trata de un parámetro de salida. Son parámetros cuyo valor es devuelto después de la ejecución el procedimiento que lo llamó.

IN OUT: Son parámetros de entrada y salida a la vez.

Tipo-de-dato: Indica el tipo de dato del parámetro (number, varchar, etc).

Procedimiento almacenado

Para ejecutar el procedimiento se utiliza la siguiente sintaxis:

```
execute nombredeprocedimiento (valor para el parámetro);

begin

Nombredeprocedimiento;

end;
```

Para eliminar un procedimiento se utiliza la siguiente sintaxis:

```
drop procedure nombredeprocedimiento;
```

Procedimiento almacenado

1. Crear un procedimiento almacenado que reciba el documento del empleado y le aumente el 10 % a sus salario.

1. Crear un procedimiento almacenado que reciba el documento del empleado y la cantidad el porcentaje del aumento del sueldo.

1. Crear un procedimiento almacenado que ingrese un nuevo empleado.