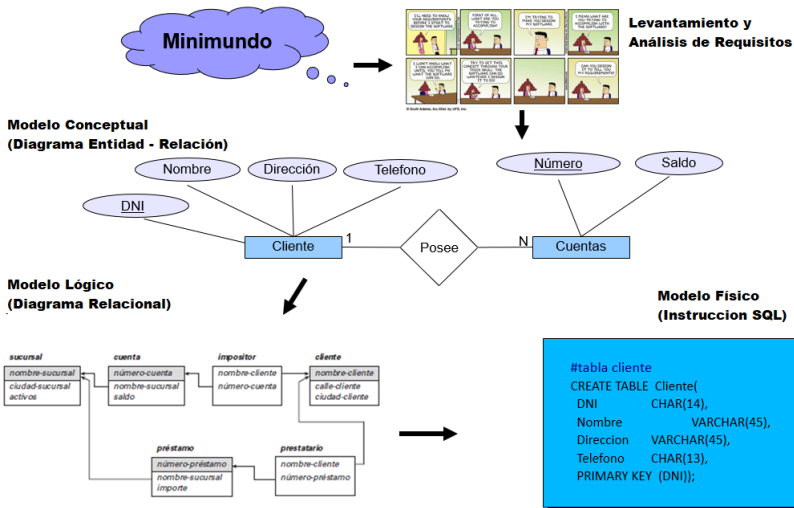


- 1 Fases del Proyecto de BD
- 2 Procedimientos Almacenados

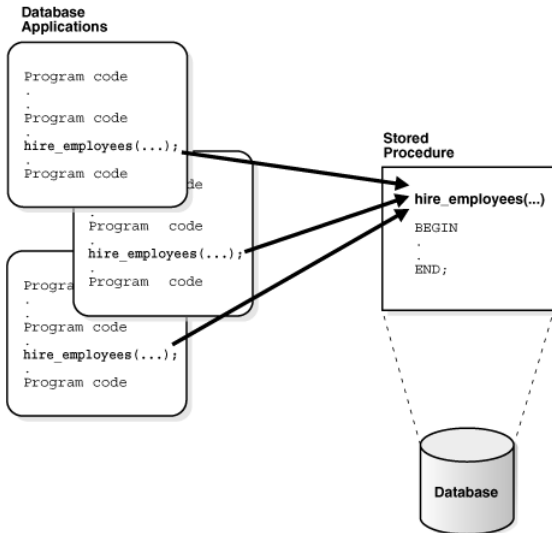
# Fases del Proyecto de BD



# Procedimientos Almacenados (Stored Procedures)

- SP es un conjunto de comandos al cual es atribuido un nombre.
- Este conjunto se encuentra almacenado en la BD y puede ser invocado en cualquier momento por el SGBD o por un sistema que utiliza la misma.
- SP son utilizados para mover gran parte de código de manipulación de datos para el servidor, ello elimina la transferencia de datos del servidor al cliente, por la red, para manipulación → reducción de tráfico de la red → mejor performance general.

# Procedimientos Almacenados (Stored Procedures)



# Procedimientos Almacenados (Stored Procedures)

- Siempre que una aplicación cliente envía un comando SQL para el servidor, el comando tiene que tener la sintaxis analizada y, a continuación, es enviado a un programa optimizador del SGBD para la formulación de un plan de ejecución.
- Los SP son analizados y optimizados una única vez (cuando son creados), presentan mejor rendimiento que los comandos SQL enviados por las aplicaciones.

# Procedimientos Almacenados (Stored Procedures)

- SP pueden reducir el tiempo de desarrollo y manutención de los sistemas pues las SP pueden ser compartidos por todas las aplicaciones existentes. El mantenimiento es sencillo porque es posible alterar un SP sin tener que alterar y/o recompilar cada aplicación cliente.
- Puede ser más veloz dado que generalmente el servidor es una de las máquinas más poderosas en la red.

# Procedimientos Almacenados - Sintaxis

```

CREATE
    [DEFINER = user]
    PROCEDURE sp_name ([proc_parameter[,...]]]
    [characteristic ...] routine_body

CREATE
    [DEFINER = user]
    FUNCTION sp_name ([func_parameter[,...]]]
    RETURNS type
    [characteristic ...] routine_body

proc_parameter:
    [ IN | OUT | INOUT ] param_name type

func_parameter:
    param_name type

type:
    Any valid MySQL data type

characteristic: {
    COMMENT 'string'
    | LANGUAGE SQL
    | [NOT] DETERMINISTIC
    | { CONTAINS SQL | NO SQL | READS SQL DATA | MODIFIES SQL DATA }
    | SQL SECURITY { DEFINER | INVOKER }
}

routine_body:
    Valid SQL routine statement
  
```

# Procedimientos Almacenados (Stored Procedures)

- Un SP es invocado con el comando CALL.
- No se puede invocar un SP en una expresión.
- Podemos utilizar las sentencias de control IF, ELSE, LOOP, WHILE, ... (Tarea)



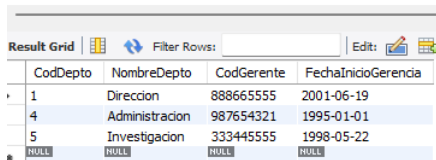
# Procedimientos Almacenados (Stored Procedures)

- Se consideran 3 tipos de parámetros (Tarea):
  - Un parámetro IN pasa un valor a un procedimiento. El procedimiento puede modificar el valor, pero la modificación no es visible para la aplicación que llama cuando el procedimiento vuelve.
  - Un parámetro OUT pasa un valor del procedimiento a la aplicación que lo llama. Su valor inicial es NULL dentro del procedimiento, y su valor es visible para la aplicación que lo llama cuando el procedimiento vuelve.
  - Un parámetro INOUT es inicializado por la aplicación que lo llama, puede ser modificado por el procedimiento, y cualquier cambio realizado por el procedimiento es visible para la aplicación que llama cuando el procedimiento regresa.

# Procedimientos Almacenados - Ejemplo 1

- Cree un SP que inserte un nuevo departamento.

- 1 • **USE** empresa;
- 2 • **SELECT** \* **FROM** departamento;



The screenshot shows a 'Result Grid' window from a database management tool. It contains a table with four columns: 'CodDeppto', 'NombreDeppto', 'CodGerente', and 'FechaInicioGerencia'. The table has five rows of data. The first three rows have numerical department codes and names, while the fourth row has 'NULL' values for all fields. The interface includes a 'Filter Rows' search bar and an 'Edit' button with a pencil icon.

	CodDeppto	NombreDeppto	CodGerente	FechaInicioGerencia
1	1	Direccion	888665555	2001-06-19
4	4	Administracion	987654321	1995-01-01
5	5	Investigacion	333445555	1998-05-22
	NULL	NULL	NULL	NULL

# Procedimientos Almacenados - Ejemplo 1

- Cree un SP que inserte un nuevo departamento.

The screenshot displays a SQL development environment. On the left, a 'SCHEMAS' pane shows the database structure, including tables like 'departamento' and 'dependiente'. The main editor window contains the following SQL code:

```
1 * USE empresa;
2
3 DELIMITER //
4 * CREATE PROCEDURE Insertar_Departamento(IN id INT, IN nomb VARCHAR(30), IN gere INT, IN fecha datetime)
5 BEGIN
6     INSERT INTO departamento VALUES (id, nomb, gere, fecha);
7 END;
8 //
9 DELIMITER ;
```

Below the code editor, an 'Output' pane shows the execution results:

Action	Time	Message
USE empresa	18:14:03	0 row(s) affected
CREATE PROCEDURE Insertar_Departamento	18:14:03	0 row(s) affected

# Procedimientos Almacenados - Ejemplo 1

- Cree un SP que inserte un nuevo departamento.

The screenshot shows a database management tool interface. On the left, the 'SCHEMAS' tree is expanded to show the 'empresaa' database, with 'Stored Procedures' selected. The 'Insertar\_Departamento' procedure is highlighted. The main editor displays the following SQL code:

```
1 • USE empresa;  
2 • CALL Insertar_Departamento(2, 'Logistica', NULL, NULL);
```

Below the editor, the 'Output' window shows the execution results:

#	Time	Action
1	18:14:39	USE empresa
2	18:14:39	CALL Insertar_Departamento(2, 'Logistica', NULL, NULL)

# Procedimientos Almacenados - Ejemplo 1

- Cree un SP que inserte un nuevo departamento.

- 1 • **USE** empresa;
- 2 • **SELECT \* FROM** departamento;

result Grid			
Filter Rows:			
Edit:			
CodDepto	NombreDepto	CodGerente	FechaInicioGerencia
1	Direccion	888665555	2001-06-19
2	Logistica	NULL	NULL
4	Administracion	987654321	1995-01-01
5	Investigacion	333445555	1998-05-22
NULL	NULL	NULL	NULL

# Procedimientos Almacenados - Ejemplo 2

- Cree un SP que devuelva el código, nombre completo y nombre de departamento de o de los empleados que comiencen con la palabra de búsqueda ingresada en su primer apellido.

- 1 • **USE** empresa;
- 2 • **SELECT** \* **FROM** empleado;

Result Grid									
Filter Rows:		Edit:		Export/Import:		Wrap Cell Content:			
CodEmpleado	Nombres	PrimerApellido	SegundoApellido	Direccion	FechaNacimiento	Sexo	Salario	Supervisor	CodDepto
123456789	Juan	Perez	Rodriguez	Calle Numero A 1	1965-01-09	M	300.00	333445555	5
333445555	Frank	Velazquez	Flores	Calle Numero B 2	1955-12-08	M	4000.00	888665555	5
453453453	Daniela	Acco	Olvarez	Calle Numero F 6	1962-07-31	F	2500.00	333445555	5
666884444	Pedro	Lima	Maldonado	Calle Numero E 5	1952-09-15	M	1200.00	333445555	5
888665555	Francisco	Linares	Gomez	Calle Numero H 8	1957-11-10	M	5500.00	NULL	1
987654321	Luisa	Santos	Ferrel	Calle Numero D 4	1951-06-20	F	430.00	888665555	4
987987987	Mateo	Vela	Marruecos	Calle Numero G 7	1979-03-29	M	2500.00	987654321	4
999887777	Alice	Jimenez	Portugal	Calle Numero C 3	1968-07-19	F	2500.00	987654321	4
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

# Procedimientos Almacenados - Ejemplo 2

- Cree un SP que devuelva el código, nombre completo y nombre de departamento de o de los empleados que comiencen con la palabra de búsqueda ingresada en su primer apellido.

```
1 • USE empresa;
2
3 DELIMITER //
4 • DROP PROCEDURE IF EXISTS Buscar_Empleado//
5 • CREATE PROCEDURE Buscar_Empleado(IN apel VARCHAR(30))
6 BEGIN
7     SELECT e.CodEmpleado 'Código',
8     CONCAT(e.PrimerApellido, ' ', e.SegundoApellido, ' ', e.Nombres) 'Nombre Completo',
9     d.NombreDepto 'Departamento'
10 FROM empleado e
11 INNER JOIN departamento d
12 ON e.CodDepto = d.CodDepto
13 WHERE UPPER(e.PrimerApellido) LIKE CONCAT(UPPER(apel) ,'%');
14 END;
15 //
16 DELIMITER ;
```

## Procedimientos Almacenados - Ejemplo 2

- Cree un SP que devuelva el código, nombre completo y nombre de departamento de o de los empleados que comiencen con la palabra de búsqueda ingresada en su primer apellido.

- 1 • **USE** empresa;
- 2 • **CALL** Buscar\_Empleado("li");

Result Grid			Filter Rows:	Export:	Wrap Cel
	Código	Nombre Completo	Departamento		
▶	888665555	Linares Gomez, Francisco	Direccion		
	666884444	Lima Maldonado, Pedro	Investigacion		



# Procedimientos Almacenados - Ejemplo 3

- Cree un SP que aumente el salario de un empleado en cierta cantidad de dinero cuando el salario sea menor o igual a 1000 soles; y cuando el salario sea mayor a 1000 soles, adicionar la mitad de la cantidad indicada.

- 1 • **USE** empresa;
- 2 • **SELECT \* FROM** empleado;

Result Grid									
Filter Rows:									
Edit: Export/Import: Wrap Cell Content: <input type="checkbox"/>									
CodEmpleado	Nombres	PrimerApellido	SegundoApellido	Direccion	FechaNacimiento	Sexo	Salario	Supervisor	CodDepto
123456789	Juan	Perez	Rodriguez	Calle Numero A 1	1965-01-09	M	300.00	333445555	5
333445555	Frank	Velazquez	Flores	Calle Numero B 2	1955-12-08	M	4000.00	888665555	5
453453453	Daniela	Acco	Olvarez	Calle Numero F 6	1962-07-31	F	2500.00	333445555	5
666884444	Pedro	Lima	Maldonado	Calle Numero E 5	1952-09-15	M	1200.00	333445555	5
888665555	Francisco	Linares	Gomez	Calle Numero H 8	1957-11-10	M	5500.00	NULL	1
987654321	Luisa	Santos	Ferrel	Calle Numero D 4	1951-06-20	F	430.00	888665555	4
987987987	Mateo	Vela	Marruecos	Calle Numero G 7	1979-03-29	M	2500.00	987654321	4
999887777	Alice	Jimenez	Portugal	Calle Numero C 3	1968-07-19	F	2500.00	987654321	4
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

# Procedimientos Almacenados - Ejemplo 3

```
1 • USE empresa;
2
3 DELIMITER //
4 DROP PROCEDURE IF EXISTS Aumentar_Sueldo//
5 CREATE PROCEDURE Aumentar_Sueldo(IN id INT, IN aumento DECIMAL(12,2))
6 BEGIN
7     DECLARE salario_ante DECIMAL(12,2);
8     DECLARE limite DECIMAL(12,2);
9
10    SET limite = 1000.00;
11    SET salario_ante = (SELECT e.salario from empleado e WHERE e.CodEmpleado = id);
12
13    IF salario_ante <= limite THEN
14        UPDATE empleado e SET e.salario = e.salario + aumento WHERE e.CodEmpleado = id;
15    ELSE
16        SET aumento = aumento / 2;
17        UPDATE empleado e SET e.salario = e.salario + aumento WHERE e.CodEmpleado = id;
18    END IF;
19 END;
20 //
21 DELIMITER ;
```

# Procedimientos Almacenados - Ejemplo 3

- 1 • **USE** empresa;
- 2 • **CALL** Aumentar\_Sueldo(123456789, 600.00);
- 3 • **SELECT** \* **FROM** empleado;

Result Grid										
Filter Rows: <input type="text"/> Edit:    Export/Import:   Wrap Cell Content:										
	CodEmpleado	Nombres	PrimerApellido	SegundoApellido	Direccion	FechaNacimiento	Sexo	Salario	Supervisor	CodDepto
	123456789	Juan	Perez	Rodriguez	Calle Numero A 1	1965-01-09	M	900.00	333445555	5
	333445555	Frank	Velazquez	Flores	Calle Numero B 2	1955-12-08	M	4000.00	888665555	5
	453453453	Daniela	Acco	Olvarez	Calle Numero F 6	1962-07-31	F	2500.00	333445555	5
	666884444	Pedro	Lima	Maldonado	Calle Numero E 5	1952-09-15	M	1200.00	333445555	5
	888665555	Francisco	Linares	Gomez	Calle Numero H 8	1957-11-10	M	5500.00	NULL	1
	987654321	Luisa	Santos	Ferrel	Calle Numero D 4	1951-06-20	F	430.00	888665555	4
	987987987	Mateo	Vela	Marruecos	Calle Numero G 7	1979-03-29	M	2500.00	987654321	4
	999887777	Alice	Jimenez	Portugal	Calle Numero C 3	1968-07-19	F	2500.00	987654321	4
	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

# Procedimientos Almacenados - Ejemplo 3

- 1 • **USE** empresa;
- 2 • **CALL** Aumentar\_Sueldo(987987987, 600.00);
- 3 • **SELECT** \* **FROM** empleado;

Result Grid										
Filter Rows:		Edit:		Export/Import:		Wrap Cell Content:				
	CodEmpleado	Nombres	PrimerApellido	SegundoApellido	Direccion	FechaNacimiento	Sexo	Salario	Supervisor	CodDepto
▶	123456789	Juan	Perez	Rodriguez	Calle Numero A 1	1965-01-09	M	900.00	333445555	5
	333445555	Frank	Velazquez	Flores	Calle Numero B 2	1955-12-08	M	4000.00	888665555	5
	453453453	Daniela	Acco	Olvarez	Calle Numero F 6	1962-07-31	F	2500.00	333445555	5
	666884444	Pedro	Lima	Maldonado	Calle Numero E 5	1952-09-15	M	1200.00	333445555	5
	888665555	Francisco	Linares	Gomez	Calle Numero H 8	1957-11-10	M	5500.00	NULL	1
	987654321	Luisa	Santos	Ferrel	Calle Numero D 4	1951-06-20	F	430.00	888665555	4
	987987987	Mateo	Vela	Marruecos	Calle Numero G 7	1979-03-29	M	2800.00	987654321	4
	999887777	Alice	Jimenez	Portugal	Calle Numero C 3	1968-07-19	F	2500.00	987654321	4
*	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL