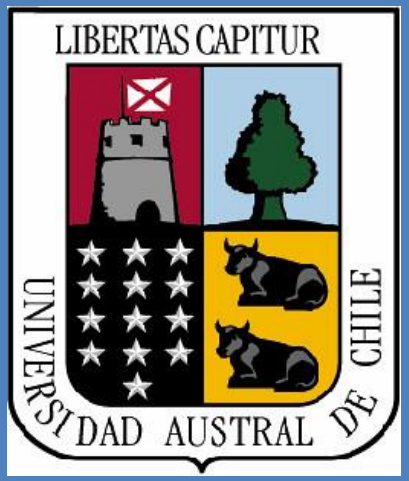
TALLER DE PROGRAMACIÓN
UNIDAD 2: BASES DE DATOS

PROFESOR GERMÁN BARRIENTOS

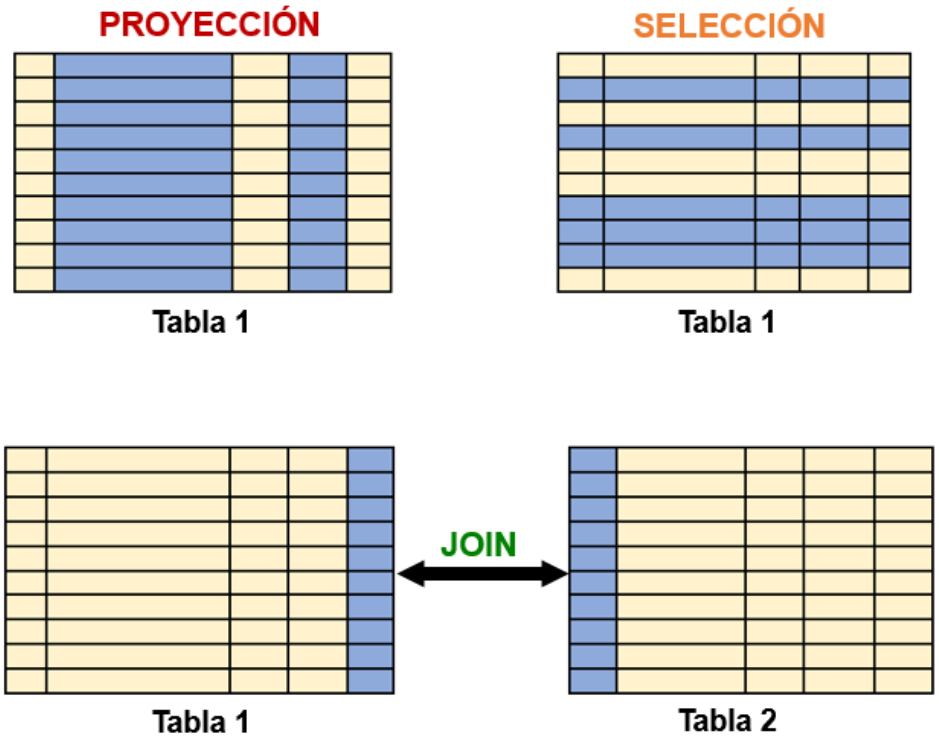
PROFESOR JOEL TORRES



QUÉ APRENDEREMOS

- Cómo utilizar sentencia SELECT básica para mostrar datos desde las Tablas de la Base de Datos.
- Cómo utilizar operadores matemáticos en una sentencia SELECT.
- Cómo unir valores/columnas/expresiones en una sentencia SELECT.
- Cómo utilizar Alias para asignar nombres lógicos a las columnas y expresiones obtenidas en una sentencia SELECT.
- Cómo mostrar la información en un orden específico.

CAPACIDADES DE LA SENTENCIA SELECT



SELECCIONANDO LAS COLUMNAS DE UNA TABLA

SELECT permite mostrar una o más columnas de las tablas, además de expresiones

Si se desea mostrar más de una columna o expresión, éstas se deben separar con comas

```
SELECT  * | { [ DISTINCT ]  columna|expresión [alias],...}
FROM   tabla
[WHERE  condición]
[ORDER  BY {columna, alias, expresión, posición_numérica} [ASC|DESC]];
```

SELECCIONANDO TODAS LAS COLUMNAS DE LA TABLA

```
SELECT *
  FROM departments;
```

	DEPARTMENT_ID	DEPARTMENT_NAME	MANAGER_ID	LOCATION_ID
1	10	Administration	200	1700
2	20	Marketing	201	1800
3	30	Purchasing	114	1700
4	40	Human Resources	203	2400
5	50	Shipping	121	1500
6	60	IT	103	1400
7	70	Public Relations	204	2700
8	80	Sales	145	2500
9	90	Executive	100	1700
10	100	Finance	108	1700
11	110	Accounting	205	1700
...	...	...	...	...
23	230	IT Helpdesk	(null)	1700
24	240	Government Sales	(null)	1700
25	250	Retail Sales	(null)	1700
26	260	Recruiting	(null)	1700
27	270	Payroll	(null)	1700

SELECCIONANDO COLUMNAS ESPECÍFICAS DE LA TABLA

```
SELECT department_id, location_id
 FROM departments;
```

	DEPARTMENT_ID	LOCATION_ID
1	10	1700
2	20	1800
3	30	1700
4	40	2400
5	50	1500
6	60	1400
7	70	2700
8	80	2500
9	90	1700
10	100	1700
11	110	1700
.....	.....	.....
24	240	1700
25	250	1700
26	260	1700
27	270	1700

```
SELECT location_id, department_id
 FROM departments;
```

	LOCATION_ID	DEPARTMENT_ID
1	1700	10
2	1800	20
3	1700	30
4	2400	40
5	1500	50
6	1400	60
7	2700	70
8	2500	80
9	1700	90
10	1700	100
11	1700	110
.....	.....	.....
24	1700	240
25	1700	250
26	1700	260
27	1700	270

SELECCIONANDO COLUMNAS ESPECÍFICAS DE LA TABLA

```
SELECT employee_id, first_name, last_name, salary
FROM employees;
```

	EMPLOYEE_ID	FIRST_NAME	LAST_NAME	SALARY
1	100	Steven	King	24000
2	101	Neena	Kochhar	17000
3	102	Lex	De Haan	17000
4	103	Alexander	Hunold	9000
5	104	Bruce	Ernst	6000
6	105	David	Austin	4800
7	106	Valli	Pataballa	4800
8	107	Diana	Lorentz	4200
9	108	Nancy	Greenberg	12008
10	109	Daniel	Faviet	9000

.....

.....

104	203	Susan	Mavris	6500
105	204	Hermann	Baer	10000
106	205	Shelley	Higgins	12008
107	206	William	Gietz	8300

USANDO OPERADORES ARITMÉTICOS

Para efectuar cálculos con los datos de las tablas se deben usar expresiones aritméticas

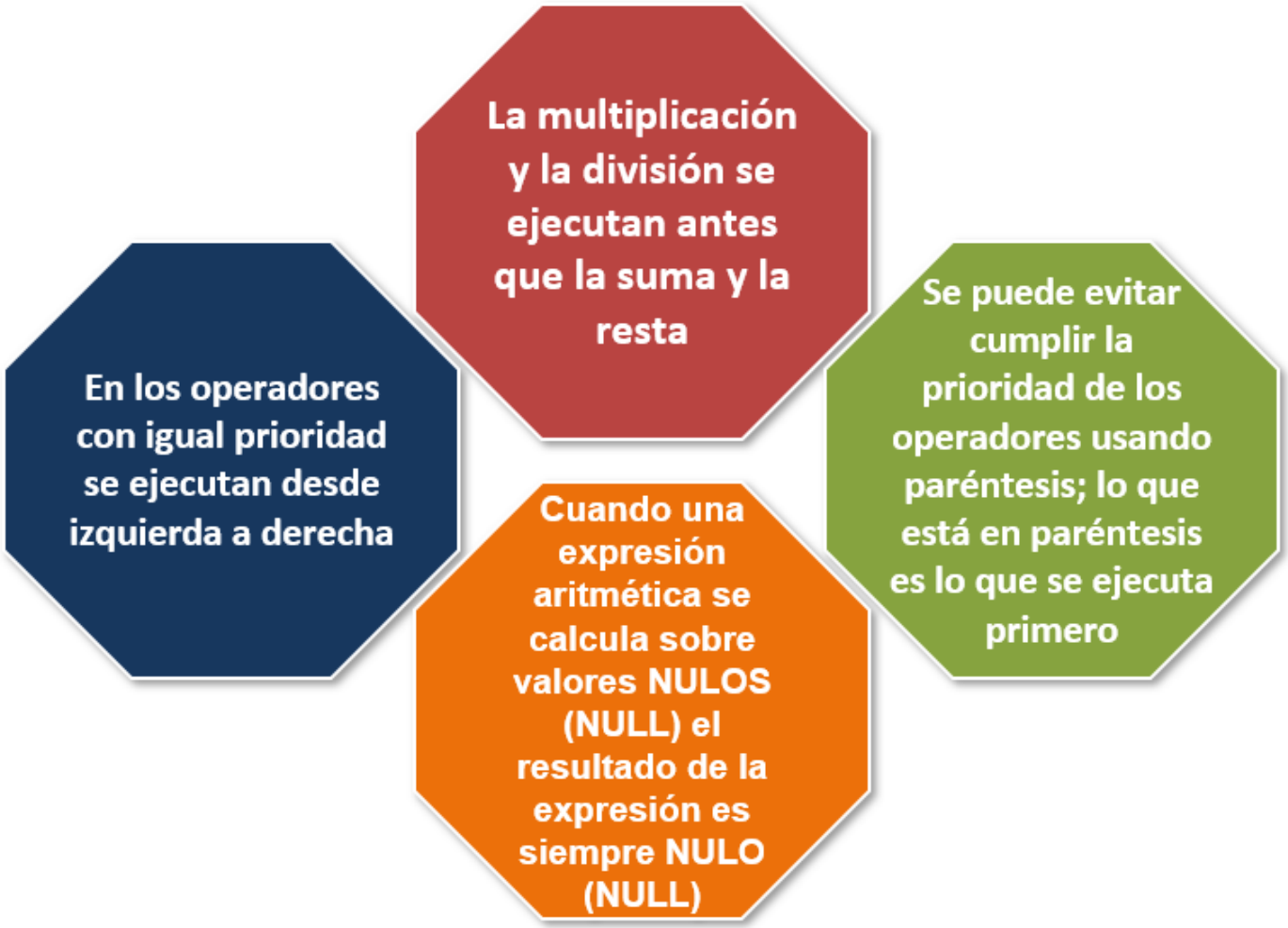
Una expresión aritmética puede contener nombre de columnas, constantes de valores numéricos y operadores aritméticos

Los operadores aritméticos se pueden usar en cualquier cláusula de una sentencia DML excepto en la cláusula FROM

- Los operadores que se pueden utilizar en una sentencia SQL son

OPERADOR	DESCRIPCIÓN
+	Suma
-	Resta
*	Multiplicación
/	División

USANDO OPERADORES ARITMÉTICOS



USANDO OPERADORES ARITMÉTICOS

```
SELECT  last_name, salary,  salary + 300
 FROM  employees;
```

	LAST_NAME	SALARY	SALARY+300
1	King	24000	24300
2	Kochhar	17000	17300
3	De Haan	17000	17300
4	Hunold	9000	9300
5	Ernst	6000	6300
6	Austin	4800	5100
7	Pataballa	4800	5100
8	Lorentz	4200	4500
9	Greenberg	12008	12308
10	Faviet	9000	9300
...	...	...	...
104	Mavris	6500	6800
105	Baer	10000	10300
106	Higgins	12008	12308
107	Gietz	8300	8600

USANDO OPERADORES ARITMÉTICOS

```
SELECT  last_name, salary, 12*salary+100
 FROM  employees;
```

	LAST_NAME	SALARY	12*SALARY+100
1	King	24000	288100
2	Kochhar	17000	204100
3	De Haan	17000	204100
4	Hunold	9000	108100
5	Ernst	6000	72100
6	Austin	4800	57700
7	Pataballa	4800	57700
8	Lorentz	4200	50500
9	Greenberg	12008	144196
10	Faviet	9000	108100
...	...	...	...
104	Mavris	6500	78100
105	Baer	10000	120100
106	Higgins	12008	144196
107	Gietz	8300	99700

DEFINIENDO ALIAS DE COLUMNAS

Renombra una columna

Debe ir entre doble comillas si posee espacios, caracteres especiales o es case-sensitive

Se muestran en lugar del nombre real de la columna o expresión

Va a continuación del nombre de columna o expresión

Útil cuando se efectúan cálculos

DEFINIENDO ALIAS DE COLUMNAS

```
SELECT last_name AS apellido , salary salario
 FROM employees;
```

	APELLIDO	SALARIO
1	King	24000
2	Kochhar	17000
3	De Haan	17000
4	Hunold	9000
5	Ernst	6000
6	Austin	4800
7	Pataballa	4800
8	Lorentz	4200
9	Greenberg	12008
10	Faviet	9000
...	...	...
104	Mavris	6500
105	Baer	10000
106	Higgins	12008
107	Gietz	8300

DEFINIENDO ALIAS DE COLUMNAS

```
SELECT  last_name, salary  "Salario" , salary* 1.25  "Salario Aumentado en 25%"
 FROM  employees;
```

	LAST_NAME	Salario	Salario Aumentado en 25%
1	King	24000	30000
2	Kochhar	17000	21250
3	De Haan	17000	21250
4	Hunold	9000	11250
5	Ernst	6000	7500
6	Austin	4800	6000
7	Pataballa	4800	6000
8	Lorentz	4200	5250
9	Greenberg	12008	15010
10	Faviet	9000	11250
11	Chen	8200	10250
...	...	...	...
...	...	...	...
105	Baer	10000	12500
106	Higgins	12008	15010
107	Gietz	8300	10375

USANDO OPERADOR DE CONCATENACIÓN

Permite unir columnas y/o cadena de caracteres literales y formas una sola columna de salida

Si se desean concatenar fechas y/o caracteres literales, estos deben ir entre comillas simples

Crea una columna resultante que es una expresión de caracteres

Si a una columna se concatena un NULO, el resultado es una expresión de tipo caracter

USANDO OPERADOR DE CONCATENACIÓN

```
SELECT  employee_id AS  "IDENTIFICACION EMPLEADO",
        CONCAT (first_name , ' ' , last_name ) AS "NOMBRE DEL EMPLEADO",
         salary salario
 FROM  employees;
```

	IDENTIFICACION EMPLEADO	NOMBRE DEL EMPLEADO	SALARIO
1	100	Steven King	24000
2	101	Neena Kochhar	17000
3	102	Lex De Haan	17000
4	103	Alexander Hunold	9000
5	104	Bruce Ernst	6000
6	105	David Austin	4800
7	106	Valli Pataballa	4800
8	107	Diana Lorentz	4200
9	108	Nancy Greenberg	12008
10	109	Daniel Faviet	9000
...	...	...	...
104	203	Susan Mavris	6500
105	204	Hermann Baer	10000
106	205	Shelley Higgins	12008
107	206	William Gietz	8300

USANDO OPERADOR DE CONCATENACIÓN

```
SELECT  CONCAT (last_name , ' pertenece al departamento ‘   ,
          department_id) AS "Detalle de Empleados"
 FROM  employees;
```

	Detalle de Empleados
1	King pertenece al departamento 90
2	Kochhar pertenece al departamento 90
3	De Haan pertenece al departamento 90
4	Hunold pertenece al departamento 60
5	Ernst pertenece al departamento 60
6	Austin pertenece al departamento 60
7	Pataballa pertenece al departamento 60
8	Lorentz pertenece al departamento 60
9	Greenberg pertenece al departamento 100
10	Faviet pertenece al departamento 100
...	...
104	Mavris pertenece al departamento 40
105	Baer pertenece al departamento 70
106	Higgins pertenece al departamento 110
107	Gietz pertenece al departamento 110

VALORES DUPLICADOS EN LAS COLUMNAS

VALORES DUPLICADOS EN LAS COLUMNAS

```
SELECT department_id
 FROM employees;
```

	DEPARTMENT_ID
1	90
2	90
3	90
4	60
5	60
6	60
7	60
8	60
9	100
10	100
.....	
.....	
104	40
105	70
106	110
107	110

```
SELECT DISTINCT department_id
FROM employees;
```

	DEPARTMENT_ID
1	50
2	40
3	110
4	90
5	30
6	70
7	(null)
8	10
9	20
10	60
11	100
12	80

ORDENANDO LAS FILAS RECUPERADAS

Las filas que retorna una query no tienen un orden definido. Para ordenarlas se debe usar
ORDER BY

La cláusula ORDER BY va al final de la sentencia SELECT y permite ordenar en forma Ascendente (defecto) o Descendente

Se puede especificar el nombre de la columna, un alias, una expresión o la posición de la columna

```
SELECT  * | { [ DISTINCT ]  columna|expresión [alias],...}
FROM   tabla
[WHERE  condición]
[ORDER  BY {columna, alias, expresión, posición_numérica} [ASC|DESC]];
```

ORDENANDO LAS FILAS RECUPERADAS

```
SELECT  last_name, job_id, hire_date
 FROM  employees
ORDER BY hire_date;
```

	LAST_NAME	JOB_ID	HIRE_DATE
1	De Haan	AD_VP	13/01/2001
2	Gietz	AC_ACCOUNT	07/06/2002
3	Baer	PR_REP	07/06/2002
4	Mavris	HR_REP	07/06/2002
5	Higgins	AC_MGR	07/06/2002
6	Faviet	FI_ACCOUNT	16/08/2002
7	Greenberg	FI_MGR	17/08/2002
8	Raphaely	PU_MAN	07/12/2002
9	Kaufling	ST_MAN	01/05/2003
10	Khoo	PU_CLERK	18/05/2003
...	...	...	...
104	Markle	ST_CLERK	08/03/2008
105	Ande	SA_REP	24/03/2008
106	Banda	SA_REP	21/04/2008
107	Kumar	SA_REP	21/04/2008

```
SELECT  last_name, job_id, hire_date
 FROM  employees
ORDER BY hire_date DESC;
```

	LAST_NAME	JOB_ID	HIRE_DATE
1	Kumar	SA_REP	21/04/2008
2	Banda	SA_REP	21/04/2008
3	Ande	SA_REP	24/03/2008
4	Markle	ST_CLERK	08/03/2008
5	Lee	SA_REP	23/02/2008
6	Philtanker	ST_CLERK	06/02/2008
7	Geoni	SH_CLERK	03/02/2008
8	Zlotkey	SA_MAN	29/01/2008
9	Marvins	SA_REP	24/01/2008
10	Grant	SH_CLERK	13/01/2008
...	...	...	...
104	Mavris	HR_REP	07/06/2002
105	Baer	PR_REP	07/06/2002
106	Higgins	AC_MGR	07/06/2002
107	De Haan	AD_VP	13/01/2001

ORDENANDO LAS FILAS RECUPERADAS

```
SELECT  employee_id, last_name, salary,
        salary*12 "Salario Anual"
 FROM  employees
ORDER BY "Salario Anual" ;
```

```
SELECT  employee_id, last_name, salary,
        salary*12 "Salario Anual"
 FROM  employees
ORDER BY salary*12;
```

	EMPLOYEE_ID	LAST_NAME	SALARY	Salario Anual
1	132	Olson	2100	25200
2	128	Markle	2200	26400
3	136	Philtanker	2200	26400
4	135	Gee	2400	28800
5	127	Landry	2400	28800
6	119	Colmenares	2500	30000
7	131	Marlow	2500	30000
8	140	Patel	2500	30000
9	144	Vargas	2500	30000
10	182	Sullivan	2500	30000

.....

.....

104	145	Russell	14000	168000
105	102	De Haan	17000	204000
106	101	Kochhar	17000	204000
107	100	King	24000	288000

TALLER DE PROGRAMACIÓN
UNIDAD 2: BASES DE DATOS

PROFESOR GERMÁN BARRIENTOS

PROFESOR JOEL TORRES