PUCP

1INF33 Base de Datos

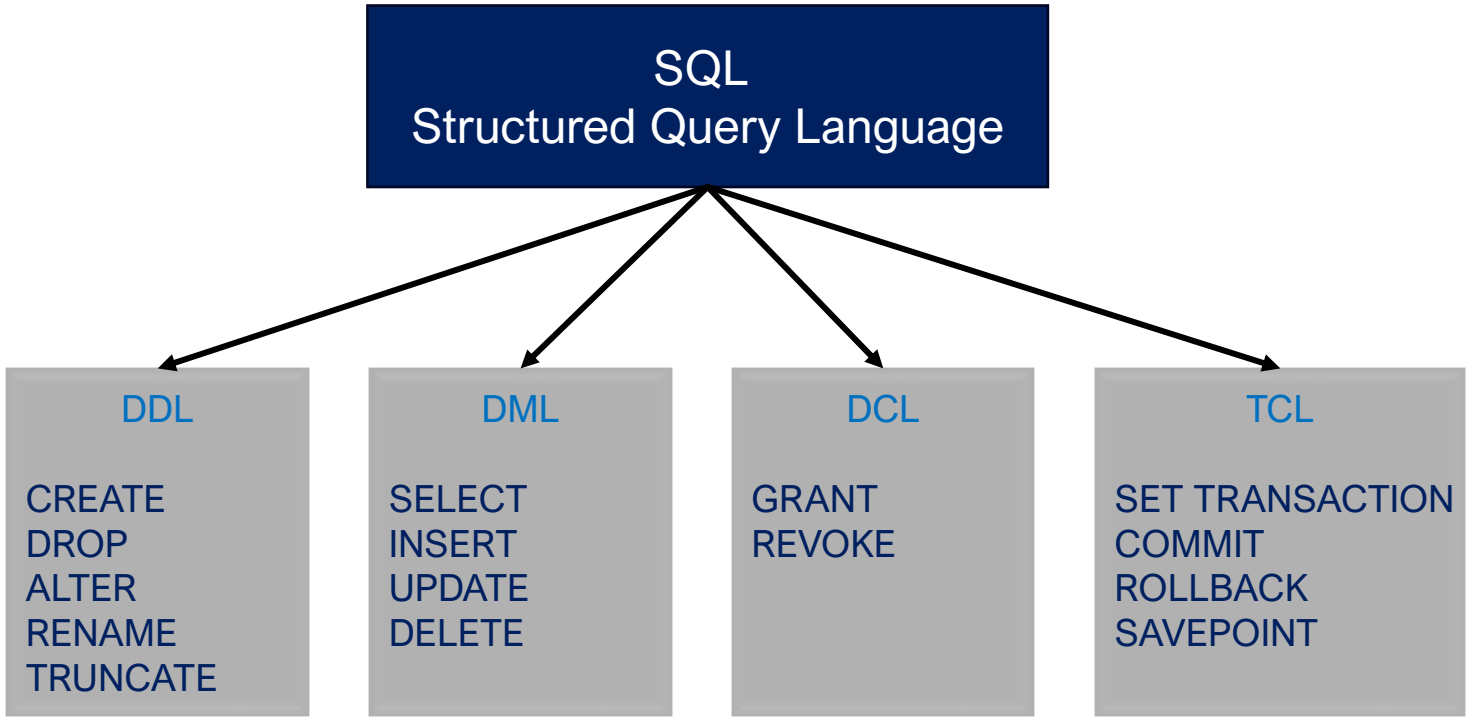
SQL-DML

2024

Profesores del curso

ÍNDICE

Antecedentes



Saberes previos

- DML, lenguaje de manipulación de datos
- Uso de SELECT
- Uso del JOIN

Operadores

Operaciones

ROWNUM: devuelve el número de la fila de una consulta. Aparece el número de cada fila en la posición de la tabla. Esta función actualiza sus valores usando subconsultas de modo que la consulta:

```
SELECT rownum as posicion, saldo, nomproveedor, razonsocial
FROM
    (SELECT saldo, nomproveedor, razonsocial
     FROM Proveedor
    ORDER BY saldo DESC )
WHERE ROWNUM <= 5;
```

Este ejemplo permite obtener a los cinco proveedores con mayor saldo actualmente.

Operaciones

Expresiones aritméticas con fechas

Desde que las bases de datos almacenan fechas como números, se pueden realizar operaciones usando operadores aritméticos como sumas y restas.

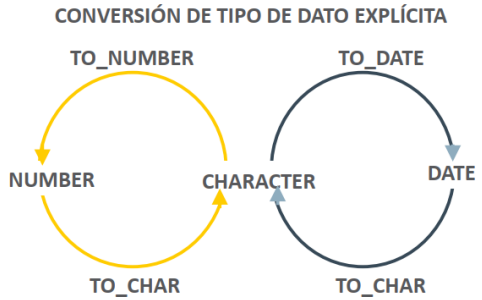
- Agregando o subtrayendo a una fecha un número de días, se obtiene una fecha: ColumnaFecha ± NúmeroDías
- Restando dos fechas entre sí, se obtiene el número de días que existe entre ambas: $ColumnaFecha_1 - ColumnaFecha_2$

En resumen, se pueden ejecutar las siguientes operaciones:

Operación	Resultado	Descripción
Fecha + numero	Fecha	Añade un número de días a una fecha
Fecha – número	Fecha	Resta un número de días de una fecha
Fecha – Fecha	Número de días	Resta una fecha a otra
Fecha + número / 24	Fecha	Añade un número de horas a una fecha

Operaciones

Expresiones aritméticas con fechas



Es posible dar formato a los valores de fecha empleando una función de conversión de caracteres y elementos de formato.

Sintaxis: **TO_CHAR**(ColumnaFecha,'ElementosFormato')

Es posible dar formato a los valores de campos numéricos empleando una función de conversión de caracteres y elementos de formato.

Sintaxis: **TO_CHAR**(ColumnaNumérica,'ElementosFormato')

El valor retornado siempre será un VARCHAR2.

Operaciones

Expresiones aritméticas con fechas

- Ejemplos:

```
SELECT id_empleado, TO_CHAR(fecha_ingreso, 'MM/YYYY') Mes_Ingreso
FROM Empleados
WHERE  primer_apellido= 'Salazar';
```

Id_empleado	Mes_Ingreso
315	07/2023

```
SELECT s.nombre, s.primer_apellido, s.email, s.telefono
FROM  Empleados s
WHERE
to_number(to_char(s.fecha_ingreso,'MM') ) = 4 and
(to_number(to_char(s.fecha_ingreso,'YYYY') ) between 2021 and 2023 );
```

Operaciones

Expresiones aritméticas con fechas

ELEMENTOS QUE DAN FORMATO A FECHAS	SINTAXIS	EJEMPLO
Día del mes (01-31)	DD	1 – 31
Día de la semana (tres primeras letras)	DY	THU
Nombre en mayúsculas del día de la semana (9 caracteres)	DAY	THUESDAY
Nombre en mayúsculas del número del día del mes	DDSPTH	TWELFTH
Mes (01 -12)	MM	01 – 12
Nombre en mayúsculas del mes (9 caracteres)	MONTH	JANUARY
Año de dos dígitos	YY	91
Año de cuatro dígitos	YYYY	2011
Horas, minutos y segundos	HH:MI:SS	09:00:00
Supresión de blancos en elementos subsiguientes	FM	

Operaciones

Operaciones con fechas y tipos de datos

- Ejemplos

  SELECT TO_CHAR(sysdate) FROM DUAL; SQL>04/02/24

  SELECT TO_CHAR(sysdate, 'dd/mm/yyyy') FROM DUAL; SQL>24/04/2024

  SELECT TO_CHAR(sysdate,'Mon') FROM DUAL; SQL>Feb

  SELECT TO_CHAR(sysdate, 'Month') FROM DUAL; SQL>Febrero

  SELECT TO_CHAR(sysdate, 'DDD') FROM DUAL; SQL>035

  SELECT TO_CHAR(sysdate, 'DD') FROM DUAL; SQL>04

- Ejemplos:

  SELECT TO_CHAR(123.456, '09999') FROM DUAL; SQL>00123

  SELECT TO_CHAR(123.456, '09999.9') FROM DUAL; SQL>00123.5

  SELECT TO_CHAR(123456, 'FM999,999,999') FROM DUAL; SQL>123,456

Funciones

Funciones

Las funciones se utilizan con operadores en un flujo de datos para permitirle crear expresiones. Tenemos

- Funciones aritméticas
- Funciones de fecha y hora
- Funciones de cadena
- Funciones de agregación

Funciones

Aritméticas

DESCRIPCIÓN	FUNCIÓN	EJEMPLO
Devuelve la potencia absoluta del valor numérico.	ABS(valor numérico)	ABS(-1)
Devuelve el entero más grande que no sea mayor que el valor numérico	FLOOR(valor numérico)	FLOOR(-1,2)
Módulo o resto	MOD(dividendo, divisor)	MOD(7,5) es 2
Raíz cuadrada	SQRT (valor numérico)	SQRT(25) es 5
Redondeo	ROUND(valor numérico, precisión)	ROUND(monto, 2) monto redondeado a 2 decimales
Truncamiento	TRUNC(valor numérico, precisión)	TRUNC(monto, 2) monto truncado a 2 decimales
Potencia	POWER(valor numérico, potencia)	POWER(monto, 3) monto elevado al cubo

Funciones

De fecha y hora

DESCRIPCIÓN	FUNCIÓN	EJEMPLO
Agregando meses	ADD_MONTHS (Columnafecha, ±Númeromeses)	ADD_MONTHS(fechaingreso,-6) devuelve 6 meses antes de la fecha de ingreso
Último día del mes de la fecha	LAST_DAY(ColumnaFecha)	LAST_DAY(fechaingreso) devuelve el último día del mes en que ingresó
Siguiente día específico de la semana después de la fecha	NEXT_DAY(ColumnaFecha, DiaSemana)	NEXT_DAY(fechaingreso, ‘ FRIDAY’) devuelve el primer viernes después de que ingresó
Número de meses entre dos fechas	MONTHS_BETWEEN(ColFecha1, ColFecha2)	MONTHS_BETWEEN(sysdate, fechaingreso) devuelve los meses trabajados
Fecha actual	SYSDATE	SYSDATE() devuelve la fecha actual

Funciones

De cadena

DESCRIPCIÓN	FUNCIÓN	EJEMPLO
Devolver la primera letra en mayúscula y el resto en minúscula	INITCAP(ColumnaCadena)	INITCAP(nombre) devuelve la primera letra de los nombres en mayúscula y el resto en minúscula
Devolver todas en mayúsculas	UPPER(ColumnaCadena)	UPPER(nombre) devuelve todos los caracteres del nombre en mayúscula
Devolver todas en minúsculas	LOWER(ColumnaCadena)	LOWER(nombre) devuelve todos los caracteres del nombre en minúscula
Devolver desde la posición dada N caracteres	SUBSTR(ColumnaCadena, Posición,N)	SUBSTR(empleo,1,3) devuelve desde la primera letra 3 caracteres del empleo
Devolver el número de caracteres de una cadena	LENGTH(ColumnaCadena)	LENGTH(nombre) devuelve el número de caracteres del nombre
El mayor entre dos valores	GREATEST(Columna1, Columna2)	GREATEST(sal,comi) devuelve el mayor entre salario y comisión
El menor entre dos valores	LEAST(Columna1,Columna 2)	LEAST('Adam','Smith') devuelve el menor: Adam

Funciones

NVL: Si la expresión o función hace referencia a una columna con un valor nulo, el resultado es también nulo. Para convertir un valor nulo en uno NO NULO, al evaluar una expresión se emplea la función NVL.

NVL evalúa una primera expresión y si NO es nula la devuelve con su valor. En caso contrario devuelve la segunda expresión.

Sintaxis: NVL (expresion1, expresion2)

Ejemplo:

```
SELECT ename, sal, comm, NVL(comm,0)  expr
FROM emp
WHERE deptno=30 AND job='SALESMAN';
```

ename	sal	comm	expr
Allen	1600	300	300
Jones	2975	500	500
Martin	1250	1400	1400
Turner	1500	0	0

Funciones

De agregación

DESCRIPCIÓN	FUNCIÓN	EJEMPLO
Devuelve el número de filas para las que una o más expresiones proporcionadas son no nulas	COUNT(valor)	COUNT(expresión)
Devuelve el número total de filas recuperadas, incluidas las filas que contienen un valor nulo	COUNT(*)	COUNT(*)
Devuelve el valor máximo del argumento	MAX(valor numérico)	MAX(valorCuota)
Devuelve el valor mínimo del argumento	MIX(valor numérico)	MIN(valorCuota)
Devolver el número de caracteres de una cadena	SUM(valor numérico)	SUM(totalVentas)
Devuelve la media de valores numéricos en una expresión	AVG(valor numérico)	AVG(precioVenta)

Consultas agrupadas

Cláusula SELECT

UNION
UNION ALL
INTERSECT
MINUS

Combinan (de acuerdo con la correspondiente operación relacional) el resultado de dos consultas en una sola. La cantidad y tipos de datos de las columnas de las consultas deben ser iguales (aunque se permiten diferentes longitudes)

GROUP BY
HAVING

Agrupa filas seleccionadas en una sola con información agregada, basándose en la(s) expresión(es).
Restringe qué filas obtenidas por el GROUP BY serán devueltas por la consulta.

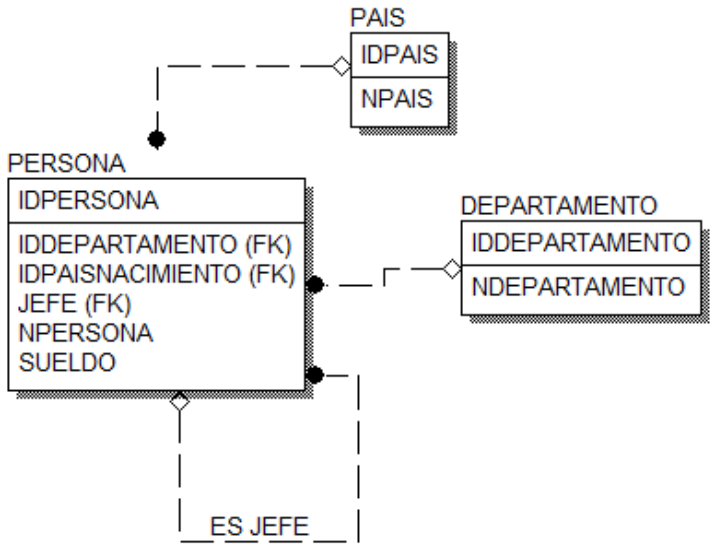
ORDER BY

Ordena las filas seleccionadas de la consulta.

Funciones

9. Obtener el sueldo mínimo, promedio y máximo de cada departamento

```
SELECT    IdDepartamento, MIN(Sueldo), AVG(Sueldo), MAX(Sueldo)
FROM      Persona
GROUP BY  IdDepartamento
```



Cláusula GROUP BY

Consultas sumarizadas

Consultas sumarizadas

Al igual que los subtotales son útiles en informes impresos, con frecuencia es conveniente sumarizar los resultados de la consulta a un nivel <<subtotal>>.

¿Cuál es el tamaño medio de un pedido?

```
SELECT AVG(IMPORTE) FROM PEDIDOS
        AVG(IMPORTE)
        ------------
          $8,256.37
```

¿Cuál es el tamaño medio de un pedido para cada vendedor?

```
SELECT REP, AVG(IMPORTE) FROM PEDIDOS  GROUP BY REP
   REP          AVG(IMPORTE)
-------         ----------
   101            $8,876.00
   102            $5,694.00
   103            $11,350.00
```

Consultas sumarizadas

La primera consulta es una consulta sumaria simple como la de los ejemplos anteriores.

La segunda consulta produce varias filas sumarias – una fila por cada grupo, sumarizando los pedidos aceptados por un solo vendedor.

SQL lleva a cabo la consulta del modo siguiente:

1. SQL divide los pedidos en grupos de pedidos, un grupo por cada vendedor. Dentro de cada grupo, todos los pedidos tienen el mismo valor en la columna REP.
2. Por cada grupo, SQL calcula el valor medio de la columna IMPORTE para todas las filas del grupo, y genera una única fila sumario de resultados. La fila contiene el valor de la columna REP del grupo y el tamaño de pedido medio calculado.

Consultas sumarizadas

He aquí algunos ejemplos adicionales de consultas agrupadas.

¿Cuál es el rango de cuotas asignadas en cada oficina?

```
SELECT OFICINA_REP, MIN(CUOTA), MAX(CUOTA) FROM REPVENTAS
GROUP BY OFICINA_REP
```

OFICINA_REP	MIN(CUOTA)	MAX(CUOTA)
NULL	NULL	NULL
11	275,000.00	300,000.00
12	200,000.00	300,000.00
13	350,000.00	300,000.00
14	350,000.00	350,000.00

Consultas sumarizadas

¿Cuántos clientes diferentes son atendidos por cada vendedor?

```
SELECT REP_CLIE, 'Clientes por rep. de venta' , COUNT(DISTINCT NUM_CLIE) FROM
CLIENTES GROUP BY REP_CLIE
```

REP_CLIE	Clientes por rep. de venta	COUNT(DISTINCT NUM_CLIE)
101	Clientes por rep. de venta	3
102	Clientes por rep. de venta	4
103	Clientes por rep. de venta	3
104	Clientes por rep. de venta	1

Cuando la cláusula GROUP BY está presente, informa a SQL que debe dividir los resultados detallados en grupos y aplicar la función de columna separadamente a cada grupo, produciendo un único resultado por cada grupo.

Consultas sumarizadas

- Múltiples columnas de agrupación

Se pueden agrupar resultados de consulta en base a contenidos de dos o más columnas.

Calcula los pedidos totales por cada vendedor (REP) y por cada cliente (CUST).

```
SELECT REP, CUST, SUM(IMPORTE) FROM PEDIDOS
GROUP BY REP, CUST
```

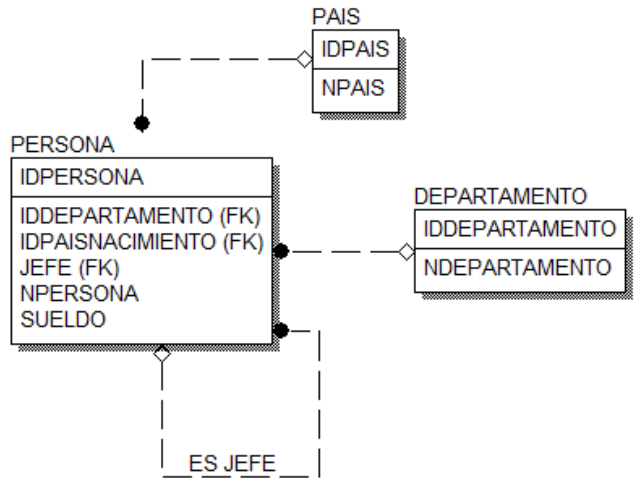
REP	CUST	SUM(IMPORTE)
101	2102	300,150.00
101	2108	20,130.00
101	2139	22,000.00
102	2106	350,000.00
102	2120	3,750.00
103	2117	4,589.00

Usando expresiones

Se desea hallar el impuesto total que se debe pagar por los sueldos, agrupados por departamento. El impuesto es el 10% del sueldo.

```
SELECT a.IdDepartamento||'-'||b.Ndepartamento Departamento,0.1*(SUM(a.Sueldo)) Impuesto
FROM Persona a, Departamento b
WHERE a.IdDepartamento = b.IdDepartamento
GROUP BY a.IdDepartamento||'-'||b.Ndepartamento;
```

Departamento	Impuesto
001-Finanzas	1500
002-Marketing	2000
...	...



Múltiples columnas de agrupación

La cláusula GROUP BY se utiliza en una sentencia SELECT para agrupar filas en un conjunto de filas resumen por valores de columnas o expresiones. La cláusula GROUP BY devuelve una fila por grupo. También se pueden ordenar los datos con la cláusula ORDER BY.

Calcular los pedidos totales para cada cliente de cada vendedor (REP), ordenados por cliente y dentro de cada cliente por vendedor.

```
SELECT CLIE, REP, SUM(IMPORTE) FROM PEDIDOS
GROUP BY CLIE, REP
ORDER BY CLIE, REP
```

CLIE	REP	SUM(IMPORTE)
2101	106	10,150.00
2102	101	23,830.00
2103	105	35,754.00

Condiciones de búsqueda de grupo

Al igual que la cláusula WHERE puede ser utilizada para seleccionar y rechazar filas individuales que participan en una consulta, la cláusula HAVING puede ser utilizada para seleccionar y rechazar grupos de filas.

La cláusula HAVING especifica por tanto una condición de búsqueda para grupos.

¿Cuál es el tamaño de pedido promedio para cada vendedor (REP) cuyos pedidos totalizan más de $30,000?.

```
SELECT REP, AVG(IMPORTE)
FROM PEDIDOS
GROUP BY REP
HAVING SUM(IMPORTE) > 30,000
```

Condiciones de búsqueda de grupo

REP	AVG(IMPORTE)
105	8,150.00
101	16,430.00
107	11,754.00
102	50,134.00

La cláusula GROUP BY dispone primero de los pedidos en grupos por vendedor. La consulta HAVING elimina entonces los grupos en donde el total de los pedidos no excede a $30,000.00.

Finalmente, la cláusula SELECT calcula el tamaño de pedido medio para cada uno de los grupos restantes y genera los resultados de la consulta.

Condiciones de búsqueda de grupo

Por cada oficina con dos o más personas (vendedores), calcular la cuota total y las ventas totales para todos los vendedores (REP) que trabajan en la oficina.

```
SELECT CIUDAD, SUM(CUOTA), SUM(REPVENTAS.VENTAS)
FROM OFICINAS, REPVENTAS
WHERE OFICINAS.OFICINA = REPVENTAS.OFICINA_REP
GROUP BY CIUDAD
HAVING COUNT(*) > 2
```

CIUDAD	SUM(CUOTA)	SUM(REPVENTAS.VENTAS)
Lima	800,150.00	735,451.00
Chiclayo	116,430.00	384,750.00
Trujillo	311,754.00	692,455.00

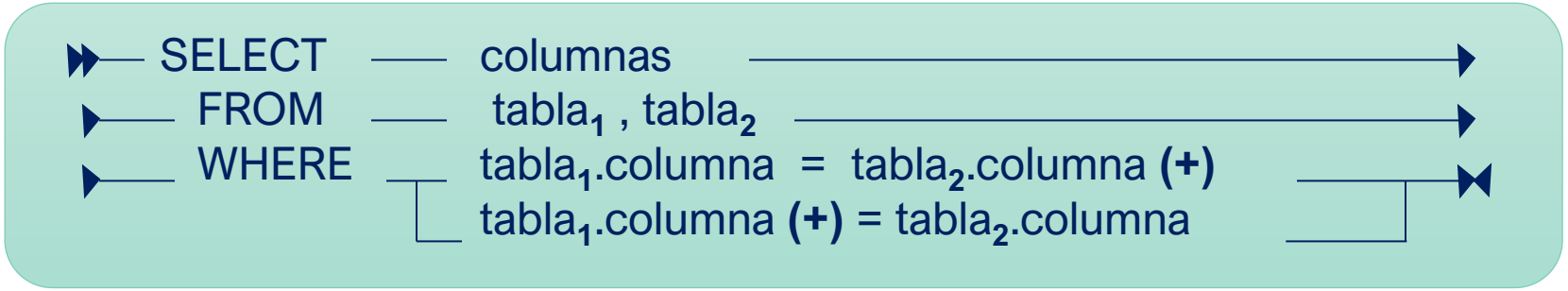
Condiciones de búsqueda de grupo

1. Se accede a las tablas OFICINAS y REPVENTAS para hallar la ciudad en donde trabaja cada Vendedor.
2. Agrupa las filas resultantes por oficinas.
3. Elimina los grupos con dos o menos filas, estás representan oficinas que no satisfacen el criterio de la cláusula HAVING.
4. Calcula la cuota total y las ventas totales para cada grupo.

Conjunción lateral

Uso de Outer join

Conjunción lateral (Outer join)



Extiende el resultado de un Join simple. No sólo retorna las filas resultado de la condición del WHERE sino además todas las de una de las tablas.

Se generan NULOS para las respectivas columnas de esta tabla (llamada “outer” o “lateral”).

No puede combinarse la misma tabla “outer” con más de una tabla en la misma consulta.

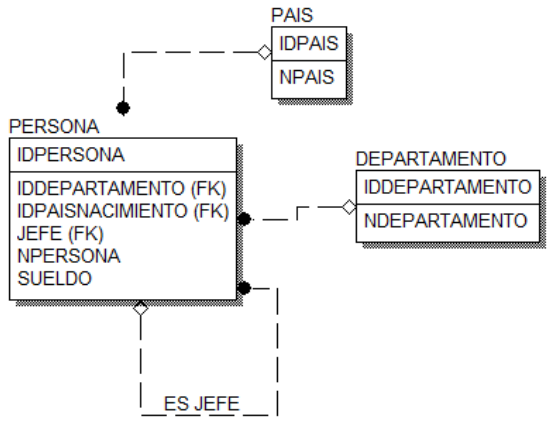
Consulta

Obtener el sueldo promedio y total de sueldos de cada departamento (aunque no tenga empleados)

Función de columna

```
SELECT  D.IdDepartamento, D.NDepartamento,
        AVG(P.Sueldo), SUM(P.Sueldo)
FROM    Departamento D, Persona P
WHERE   D.IdDepartamento (+) = P.IdDepartamento
GROUP BY        D.IdDepartamento, NDepartamento
ORDER BY        D.IdDepartamento
```

Outer join



Consulta

Obtener el sueldo promedio y total de sueldos de cada departamento (aunque no tenga empleados)

```
SELECT  D.IdDepartamento, D.NDepartamento,
        AVG(P.Sueldo), SUM(P.Sueldo)
FROM    Departamento D, Persona P
WHERE   D.IdDepartamento (+) = P.IdDepartamento
GROUP BY        D.IdDepartamento, NDepartamento
ORDER BY        D.IdDepartamento
```

El uso de OUTER JOINT, fuerza a que aparezcan valores y los completa con valores nulos.

El **(+)** se debe poner a continuación de la columna cuya tabla va a tener añadida la fila NULL imaginaria.

Condición lateral (Outer join)

Se tienen dos tablas una con datos de nombres de chicas y la ciudad donde viven y otra de chicos con los mismos campos.

Nombre	Ciudad
María	Boston
Nancy	NULL
Susan	Chicago
Beatriz	Chicago
Ana	Denver

Nombre	Ciudad
Juan	Boston
Henry	Boston
Jorge	NULL
Sam	Chicago
James	Dallas

A partir de estos datos vamos a solicitar algunas consultas.

Consulta: Outer join

- Lista las chicas y chicos de la misma ciudad y las chicas desparejadas

```
SELECT * FROM CHICAS A, CHICOS B
WHERE A.CIUDAD = B.CIUDAD (+)
```

A.Nombre	A.Ciudad	B.Nombre	Ciudad
María	Boston	Juan	Boston
María	Boston	Henry	Boston
Susan	Chicago	Sam	Chicago
Beatriz	Chicago	Sam	Chicago
Ana	Denver	NULL	NULL
Nancy	NULL	NULL	NULL

La consulta produce seis filas de resultado, mostrando los pares chico y chica emparejados y las chicas desparejadas. Los chicos desparejados faltan en los resultados.

Consulta: Outer join

- Lista las chicas y chicos de la misma ciudad y los chicos desparejados.

```
SELECT * FROM CHICAS A, CHICOS B
WHERE A.CIUDAD (+) = B.CIUDAD
```

A.Nombre	A.Ciudad	B.Nombre	Ciudad
María	Boston	Juan	Boston
María	Boston	Henry	Boston
Susan	Chicago	Sam	Chicago
Beatriz	Chicago	Sam	Chicago
NULL	NULL	Jorge	Dallas
NULL	NULL	James	NULL

La consulta también produce seis filas de resultado, mostrando los pares entre chico y chica coincidentes y los chicos no coincidentes. Esta vez las chicas desparejadas faltan en los resultados.

Conclusiones

En esta sesión, debe haber comprendido lo siguiente:

- Uso de GROUP BY y de HAVING.
- La cláusula GROUP BY se utiliza a menudo con funciones de agregación como AVG(), COUNT(), MAX(), MIN() y SUM()
- Uso del OUTER JOIN

Referencias

- AR. Elmasri y S.B. Navathe. (2007). Fundamentos de Sistema de Base de Datos, 5ta edición
- Oracle Help Center. (19 de septiembre de 2024). *SQL Language Reference*. https://docs.oracle.com/en/database/oracle/oracle-database/23/cncpt/sql.html#GUID-DA48618A-A6BB-421A-A10A-02859D8ED9AD

¡Gracias!

PUCP