Diseño y admon de Bases de Datos

Ingeniería de Sistemas
Semestre 7

Anívar Chaves Torres, 2021

Álgebra relacional

Es un lenguaje de consulta de bases de datos

Consta de un conjunto de operaciones unarias y binarias que operan sobre las relaciones que conforman la base de datos.

Cada operación se aplica sobre una o dos relaciones y da como resultado una nueva relación

Álgebra relacional

Operaciones básicas:

Proyección,
Selección,
Unión,
Diferencia,
Producto cartesiano y
Renombramiento

Operaciones avanzadas:

Intercepción,
Reunión,
División y
Asignación.

Proyección

$\Pi_{atrib1,\ atrib2}$ (Relación)

Genera una nueva relación con los atributos que se coloquen como argumentos.

Se utiliza para extraer un subconjunto de atributos de una relación.

Por defecto, el resultado de esta operación se muestra en pantalla

Proyección

$\Pi_{atrib1,\ atrib2}$ (Relación)

Ejemplo:

Estudiante(codigo, nombre, apellido, dirección, telefono)

$\Pi_{codigo,\ nombre,\ apellido}$ (Estudiante)

Resultado: (codigo, nombre, apellido)

Proyección

En PostgrSQL

$\Pi_{atrib1,\ atrib2}$ (Relación) = **select** atrib1, atrib2 **from** Relación

Ejemplo 1:

Estudiante(codigo, nombre, apellido, dirección, telefono)

$\Pi_{codigo,\ nombre,\ apellido}$ (Estudiante)

Select codigo, nombre, apellido **from** Estudiante;

Proyección

En PostgrSQL

$\Pi_{atrib1, atrib2}$ (Relación) = **select** atrib1, atrib2 **from** Relación

Ejemplo 2:

Libro(ISBN, titulo, autor, editorial, ciudad, anio)

$\Pi_{titulo, autor, anio}$ (Libro)

Select titulo, autor, anio **from** Libro;

Selección

$\sigma_{condición}$ (Relación)

Se representa con sigma en minúscula: **σ**

Genera una nueva relación con las tuplas que cumplen un criterio o condición.

Se utiliza para obtener un subconjunto de tuplas de la relación.

Es similar a aplicar un filtro a un conjunto de datos en una hoja de cálculo.

Selección

$\sigma_{condición}$ (Relación)

Ejemplo:

Factura(numfactura, fecha, valor, cedulacliente)

$\sigma_{cedulacliente = 1234}$ (Factura)

Resultado: todas las facturas del cliente con cedula 1234

Selección

$\sigma_{condición}$ (Relación)

Se utiliza en combinación con proyección

$\Pi_{atributos}$ ($\sigma_{condición}$ (Relación))

Ejemplo:

Factura(numfactura, fecha, valor, cedulacliente)

$\Pi_{numfactura,\ fecha,\ valor}$ ($\sigma_{cedulacliente\ =\ 1234}$ (Factura))

Selección

En PostgreSQL:

$\sigma_{condición}$ (Relación) = where condición

Ejemplo 1: en combinación con proyección

Factura(numfactura, fecha, valor, cedulacliente)

$\Pi_{numfactura,\ fecha,\ valor}$ ($\sigma_{cedulacliente\ =\ 1234}$ (Factura))

```
Select numfactura, fecha, valor
from Factura
where cedulacliente =  1234;
```

Selección

Ejemplo 2:

Calificacion(idestudiante, codcurso, nota, periodoacad)

$\Pi_{\text{idestudiante, codcurso, nota}}\ (\sigma_{\text{periodoacad = '2020-1'}}\ (\text{Calificacion}))$

Select idestudiante, codcurso, nota from calificacion
where periodoacad = '2020-1';

$\Pi_{\text{codcurso, nota, periodoacad}}\ (\sigma_{\text{idestudiante = 1001}}\ (\text{Calificacion}))$

Select codcurso, nota, periodoacad from calificacion
where idestudiante = 1001;

Unión

$\Pi_{atributos}$ (Relación1) U $\Pi_{atributos}$ (Relación2)

Similar a la unión de conjuntos

Genera una nueva relación con las tuplas de dos relaciones compatibles

Relaciones compatibles:

1. Las dos relaciones son del mismo grado (igual numero de atributos)
2. Los atributos tiene el mismo dominio, en el mismo orden

Unión

$\Pi_{atributos}$ (Relación1) U $\Pi_{atributos}$ (Relación2)

Ejemplo:
Deseamos una lista de estudiantes y profesores

Unir consulta a relación estudiante con consulta a relación profesor
R1 = $\Pi_{codest, nomest, apeest}$ (estudiante)
R2 = $\Pi_{codprof, nomprof, apeprof}$ (profesor)

R1 U R2
$\Pi_{codest, nomest, apeest}$ (estudiante) U $\Pi_{codprof, nomprof, apeprof}$ (profesor)

Unión

$\Pi_{atributos}$ (Relación1) U $\Pi_{atributos}$ (Relación2)

En postgreSQL: **union**

Ejemplo:

R1 = $\Pi_{codest, nomest, apeest}$ (estudiante)

Select codest, nomest, apeest from estudiante;

R2 = $\Pi_{codprof, nomprof, apeprof}$ (profesor)

Select codprof, nomprof, apeprof from profesor;

Unión

$\Pi_{atributos}$ (Relación1) U $\Pi_{atributos}$ (Relación2)

R1 U R2

$\Pi_{codest,\ nomest,\ apeest}$ (estudiante) U $\Pi_{codprof,\ nomprof,\ apeprof}$ (profesor)

En SQL

```
Select codest, nomest, apeest from estudiante
union
Select codprof, nomprof, apeprof from profesor;
```

Diferencia

Se representa con el signo menos (-): R1 - R2

Calcula la diferencia entre el conjunto de tuplas de R1 y el conjunto de tuplas R2

R1 y R2: compatibles

$\Pi_{atributos}$ (Relación1) - $\Pi_{atributos}$ (Relación2)

Muestra las tuplas de Relación1 que no están en Relación2

Diferencia

Ejemplo:

R1: estudiantes que tienen nota del curso Bases de datos

R2: estudiantes que tienen notas del curso Sistemas operacionales

R1 - R2: estudiantes que tienen nota de Bases de datos, pero no tienen nota de Sistemas operacionales

R1: $\Pi_{\text{idestudiante}} (\sigma_{\text{codcurso} = 2003} (\text{Calificacion}))$

R2: $\Pi_{\text{idestudiante}} (\sigma_{\text{codcurso} = 2005} (\text{Calificacion}))$

Diferencia

Ejemplo:

R1 – R2

$\Pi_{codes}\ (\sigma_{codcurso\ =\ 2003}\ (Calificacion)) - \Pi_{codest}\ (\sigma_{codcurso\ =\ 2005}\ (Calificacion))$

Diferencia

En postgres: **except**

R1 - R2 → R1 except R2

Ejemplo:

$\Pi_{codes}\ (\sigma_{codcurso = 2003}\ (Calificacion)) - \Pi_{codest}\ (\sigma_{codcurso = 2005}\ (Calificacion))$

```
Select codest from calificacion where codcurso = 2003
except
Select codest from calificacion where codcurso = 2005;
```

Producto cartesiano

Símbolo: X

R1 X R2

Resultado: una relación

Características del resultado:

a) Grado = Grado de R1 + Grado de R2 (sumatoria del numero de atributos)

b) Cardinalidad = cardinalidad de R x cardinalidad de R2

Producto cartesiano

Sean las relaciones:

A (a1, a2, a3)

B (b1, b2)

A X B = (a1, a2, a3, b1, b2)

Producto cartesiano

A		
a1	**a2**	**a3**
da11	da12	da13
da21	da22	da23
da31	da32	da33

B	
b1	**b2**
db11	db12
db21	db22

A X B				
a1	**a2**	**a3**	**b1**	**b2**
da11	da12	da13	db11	db12
da11	da12	da13	db21	db22
da21	da22	da23	db11	db12
da21	da22	da23	db21	db22
da31	da32	da33	db11	db12
da31	da32	da33	db21	db22

Producto cartesiano

Ejemplo:

estudiante(codest, nomest)
curso(codcur, nomcur)
calificacion(codest, codcur, peracad, nota)

Estudiante X curso

Resultado (codest, nomest, codcur, nomcur)

Tuplas = producto de: tuplas de estudiante * tuplas de curso

Producto cartesiano

En postgres: **cross join**

estudiante(codest, nomest)
curso(codcur, nomcur)
calificacion(codest, codcur, peracad, nota)

Álgebra relcional: estudiante X curso

SQL: estudiante **cross join** curso

Producto cartesiano

Ejemplo en postgreSQL:

estudiante(codest, nomest)

```
ejemplo1=# select * from estudiante;
 codest |        nomest
--------+----------------------
   1001 | Pedro Perez
   1002 | Marcela Burbano
(2 rows)
```

curso(codcur, nomcur)

```
ejemplo1=# select * from curso;
 codcur |         nomcur
--------+----------------------
   2001 | Bases de datos
   2002 | Sistemas operativos
(2 rows)
```

Producto cartesiano

Estudiante X curso

En postgreSQL: Select * from estudiante **cross join** curso:

```
ejemplo1=# select * from estudiante cross join curso;
 codest |         nomest          | codcur |         nomcur
--------+-------------------------+--------+-------------------------
   1001 | Pedro Perez             |   2001 | Bases de datos
   1001 | Pedro Perez             |   2002 | Sistemas operativos
   1002 | Marcela Burbano         |   2001 | Bases de datos
   1002 | Marcela Burbano         |   2002 | Sistemas operativos
(4 rows)
```

Resultado: todos contra todos

Producto cartesiano

La operación producto cartesiano calcula el producto de los conjuntos de tuplas sin tener en cuenta si las tablas están relacionadas o no.

El resultado del producto cartesiano es útil cuando las tablas están relacionadas mediante clave principal – clave foránea

Para obtener únicamente las tuplas que están relacionadas es necesario utilizar conjuntamente con producto cartesiano la operación selección (where)

Producto cartesiano

Consulta sobre tablas relacionadas

estudiante(codest, nomest)

calificacion(codest, codcur, peracad, nota)

```
ejemplo1=# select * from estudiante;
 codest |        nomest
--------+----------------------
   1001 | Pedro Perez
   1002 | Marcela Burbano
(2 rows)
```

Clave principal

```
ejemplo1=# select * from calificacion;
 codest | codcur |  peracad   | nota
--------+--------+------------+------
   1001 |   2001 | 2020-1     |  4.3
   1001 |   2002 | 2020-1     |  3.3
   1002 |   2002 | 2020-1     |  3.6
(3 rows)
```

Clave foránea

Producto cartesiano

$\Pi_{atributos}$ (estudiante X calificacion)

Select estudiante.codest, nomest, apeest, nota
From estudiante **cross join** calificacion;

```
ejemplo1=# select * from estudiante cross join calificacion;
 codest |        nomest         | codest | codcur |  peracad   | nota
--------+-----------------------+--------+--------+------------+------
   1001 | Pedro Perez           |   1001 |   2001 | 2020-1     |  4.3
   1002 | Marcela Burbano       |   1001 |   2001 | 2020-1     |  4.3
   1001 | Pedro Perez           |   1001 |   2002 | 2020-1     |  3.3
   1002 | Marcela Burbano       |   1001 |   2002 | 2020-1     |  3.3
   1001 | Pedro Perez           |   1002 |   2002 | 2020-1     |  3.6
   1002 | Marcela Burbano       |   1002 |   2002 | 2020-1     |  3.6
(6 rows)
```

Clave principal

Clave foránea

Operaciones conjuntas

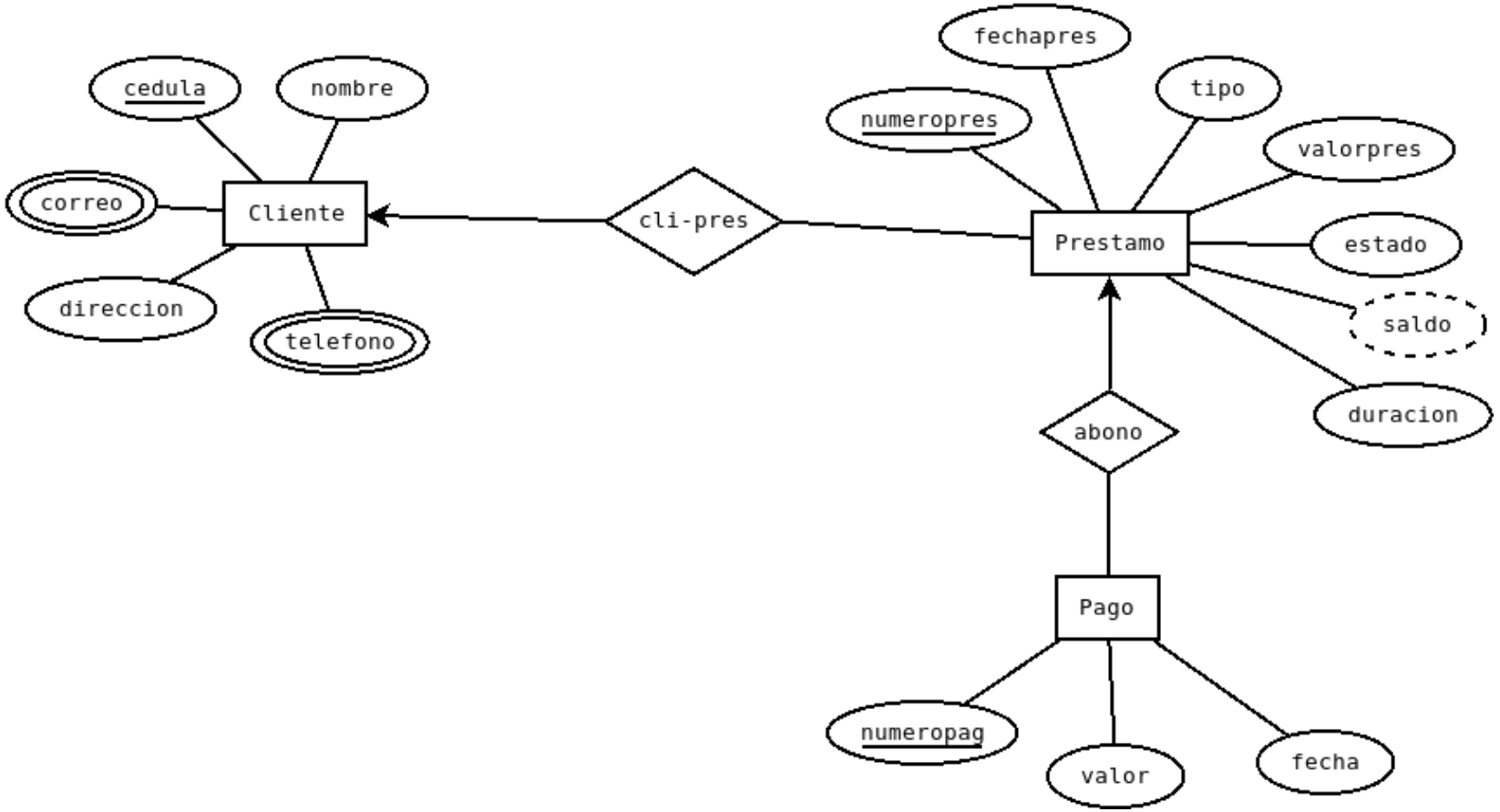
$\Pi_{atributos}\ (\sigma_{condicion}\ (R1\ X\ R2))$

Select estudiante.codest, nomest, apeest, nota
From estudiante **cross join** calificacion
Where estudiante.codest = calificacion.codest;

```
acadnf2=# Select estudiante.codest, nomest, apeest, nota
acadnf2-# From estudiante cross join calificacion
acadnf2-# Where estudiante.codest = calificacion.codest;
 codest |     nomest      |     apeest      | nota
--------+-----------------+-----------------+------
   1001 | DAYANA MARCELA  | RIVERA MARTINEZ |  4.3
   1005 | BLEIDY YURANI   | ROSERO STRIBA   |  2.7
(2 filas)
```

Producto cartesiano

Ejemplo 2: base de datos para una entidad financiera



Producto cartesiano

Esquema relacional:

Cliente (cedula, nombre, direccion)
Prestamo (numeropres, fechapres, tipo, valorpres, duracion, estado, saldo, cedula)
Pago (numeropag, valorpag, fechapag, numeropres)
Telefono (cedula, telefono)
correo (cedula, correo)

Producto cartesiano

$\Pi_{atributos}\ (\sigma_{condicion}\ (R1\ X\ R2))$

$\Pi_{cedula,\ nombre,\ fechapres,\ valorpres}\ (\sigma_{cliente.cedula\ =\ prestamo.cedula}\ (cliente\ X\ prestamo))$

SQL

Select cliente.cedula, nombre, fechapres, valorpres
From cliente **cross join** prestamo
Where cliente.cedula = prestamo.cedula;

Reunión

Esta operación genera un producto cartesiano, luego realiza la selección aplicando la condición especificada (on) y luego elimina las tuplas repetidas.

Símbolo: |X| SQL: [tipo] + join

Ejemplos:

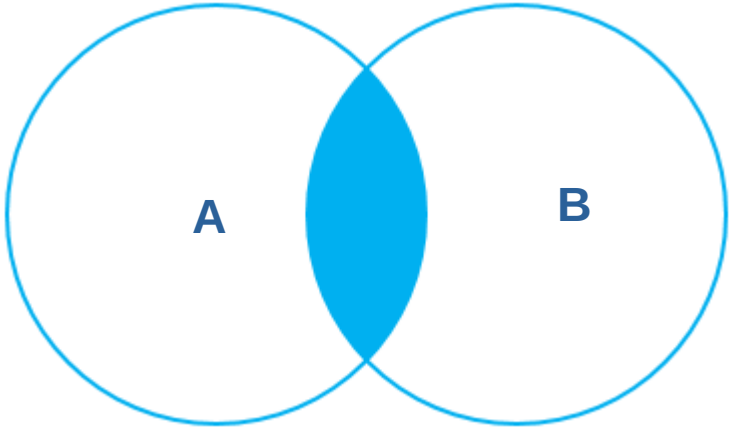
$\Pi_{idest, nomest, apeest, telefono}$ (estudiante |X| estudiantetelefono)

$\Pi_{isbn, titulo, autor}$ (libro X autor)

Reunión

Tipos de reunión

Inner join: devuelve únicamente las tuplas cuyo valor del campo en la tabla A sea igual al valor del campo relacionado en la tabla B

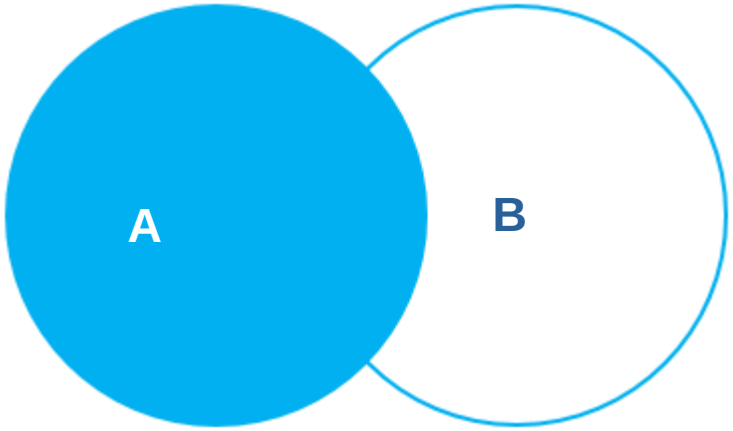



```
Select <atributos>
From B inner join B on A.id = B.id;
```

Reunión

Tipos de reunión

left join: devuelve todas las tuplas de la tabla A, con los datos de la tabla B para aquellas tuplas cuyos atributos están relacionados

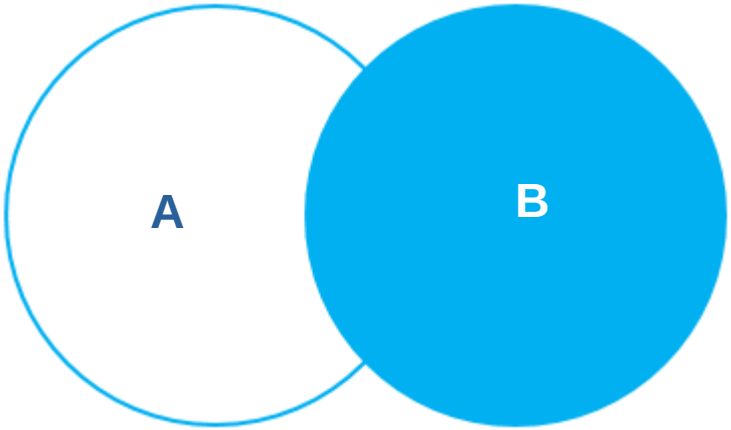



```
Select <atributos>
From B left join B on A.id = B.id;
```

Reunión

Tipos de reunión

Right join: devuelve todas las tuplas de la tabla B, con los datos de la tabla A para aquellas tuplas cuyos atributos están relacionados

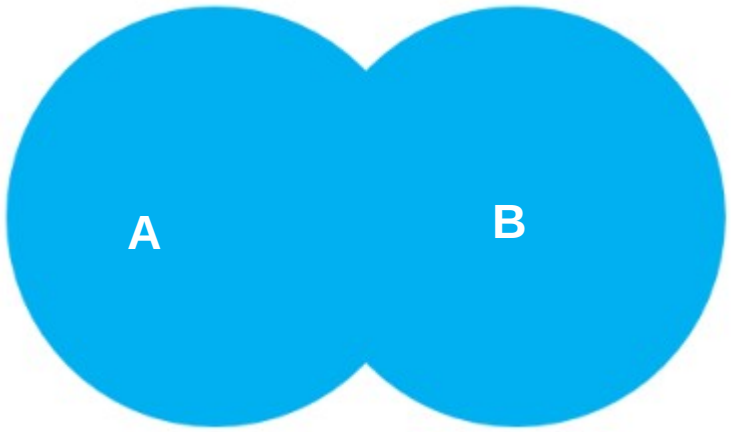



```
Select <atributos>
From B right join B on A.id = B.id;
```

Reunión

Tipos de reunión

Full join: Relaciona las tablas A y B por el valor de un atributo y muestra las tuplas relacionadas. Las tuplas no relacionadas las muestra al final.




```
Select <atributos>
From B full join B on A.id = B.id;
```

Renombramiento

Símbolo: letra griega rho minúscula

ρ_x (E)

Donde x es el nuevo nombre que se asigna a la expresión E

E puede ser el nombre de una relación o el nombre de un atributo

SQL

Para asignar alias (nuevo nombre): **as**

Renombramiento

Símbolo: letra griega rho minúscula

ρ_x (E)

Ejemplos:

Renombrar atriburo

$\rho_{identificacion}$ (idest)

Select idest as identificacion ...

Renombrar relación (tabla)

ρ_e (estudiante)

Select e.idest, nomest, apeest
From estudiante **as e**

Renombramiento

Ejemplos en SQL:

Renombrando atributos

```
select estudiante.idest as identificacon, nomest as nombre, apeest as apellido, telefono
from estudiante left join estudiantetelefono on estudiante.idest = estudiantetelefono.idest;
```

Renombrando de atributos y relaciones

```
select e.idest as identificacon, nomest as nombre, apeest as apellido, telefono
from estudiante as e left join estudiantetelefono as t on e.idest = t.idest;
```

Intersección

$R1 \cap R2$

Devuelve una relación con las tuplas que están tanto en R1 como en R2

R1 y R2 deben ser compatibles

Ejemplo:

$\Pi_{idest} (\sigma_{curso = Fisica} (calificacion) \cap \Pi_{idest} (\sigma_{curso = Calculo} (Calificacion)$

Intersección

$R1 \cap R2$

SQL: intersect

Ejemplo: estudiantes matriculados a dos cursos

$\Pi_{idest} (\sigma_{curso = Fisica} (calificacion) \cap \Pi_{idest} (\sigma_{curso = Calculo} (Calificacion)$

```
select idest from calificacion where codcur='2001'
intersect
select idest from calificacion  where codcur='2002';
```

Gracias