# Seguridad en BD

Universidad de los Andes
Demián Gutierrez
Enero 2009

# ¿Seguridad?

La seguridad informática consiste en asegurar que los ***recursos de sistemas y de información*** (material informático o programas, bases de datos, etc.) de una organización sean ***utilizados de la manera en que se decidió*** y que el acceso a la información allí contenida, así como su modificación, ***sólo sea posible a las personas que se encuentren acreditadas y dentro de los límites de su autorización***

***Fuente:*** http://es.wikipedia.org/wiki/Seguridad_informática

UNIVERSIDAD DE LOS ANDES VENEZUELA

***Físico:***

Los servidores (o nodos de cómputo) donde está el SGBD deben estar protegidos frente al acceso de extraños

***Humano:***

El personal encargado de los servidores debe ser calificado y de confianza
(sobornos / ingeniería social)

***Red:***

La red en la que opera el SGBD debe tener las protecciones correspondientes, protección en el envío de datos, firewalls, cifrado, entre otras

***Sistema Operativo:***
El sistema operativo debe estar actualizado y se deben realizar todos los esfuerzos necesarios para que no sea vulnerable

***Software / Aplicación:***
Las aplicaciones cliente de los SGBD deben ser ***diseñadas y desarrolladas*** con los niveles de seguridad adecuados

***SGBD:***
La base de datos debe ser configurada con roles, privilegios y permisos adecuados para evitar los accesos malintencionados

UNIVERSIDAD DE LOS ANDES VENEZUELA

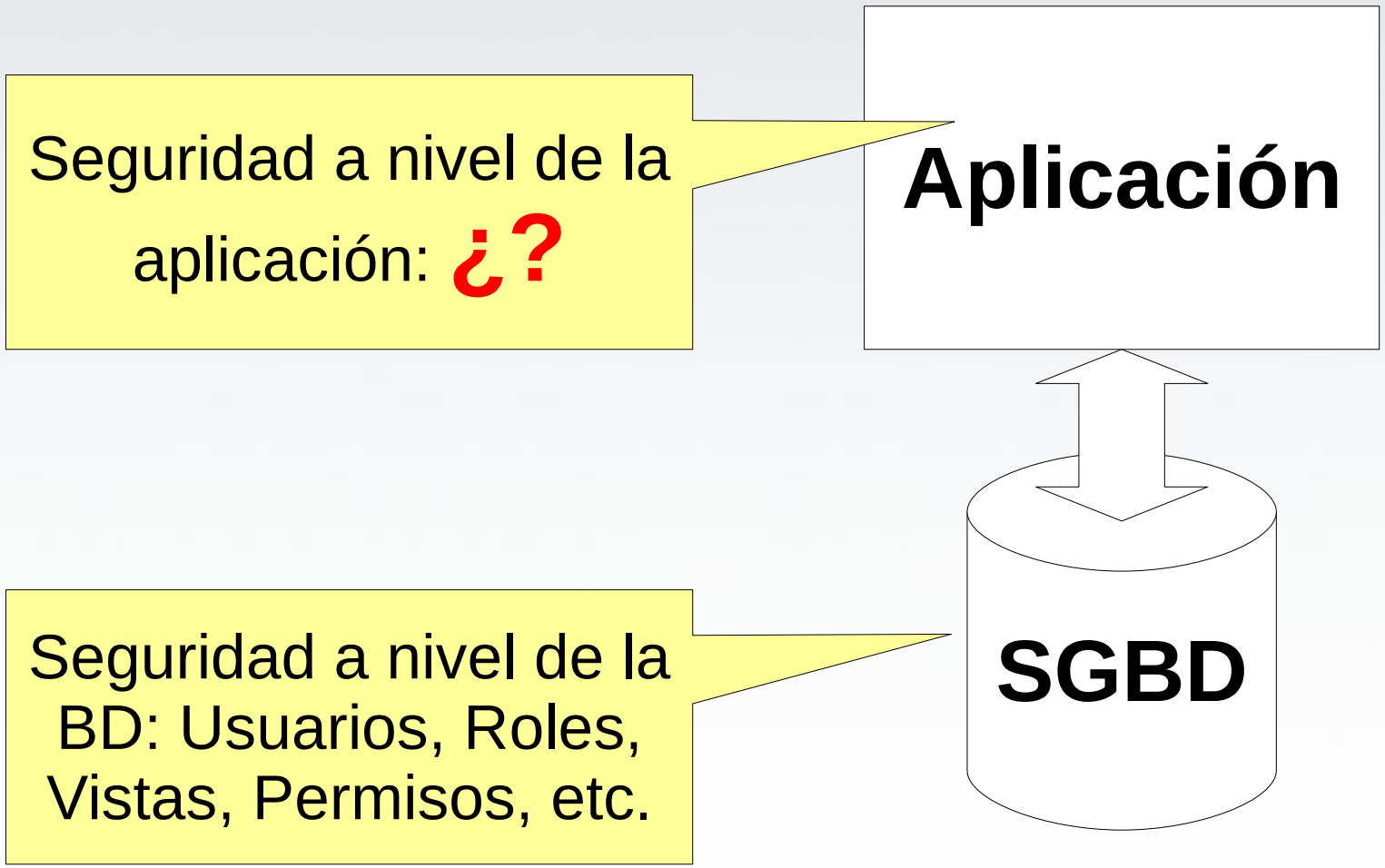
**¿Dónde implementar la seguridad?**

**¿Seguridad a nivel del SGBD o seguridad a nivel de la Aplicación?**

**Recuerden el dilema:**

**¿Validar a nivel del SGBD, a nivel de la aplicación, a nivel del cliente?**

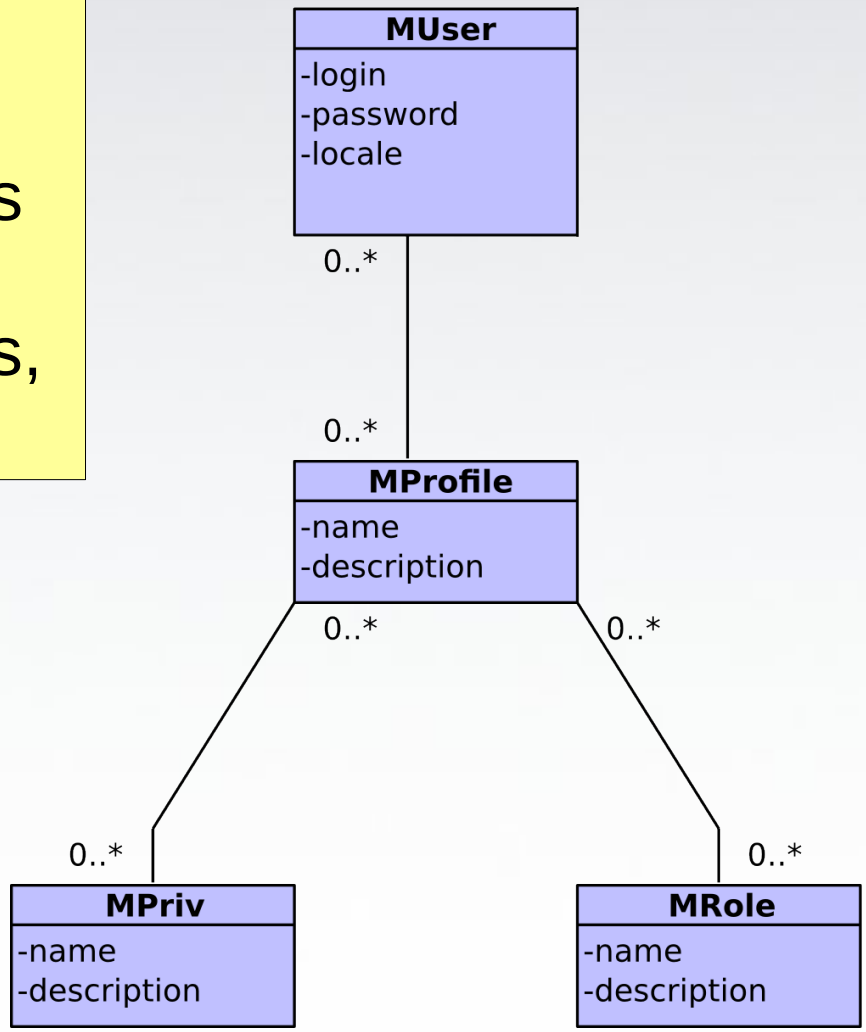
# Seguridad Obligatoria (¿SGBD que lo soporten?)



# Seguridad Obligatoria (¿SGBD que lo soporten?)

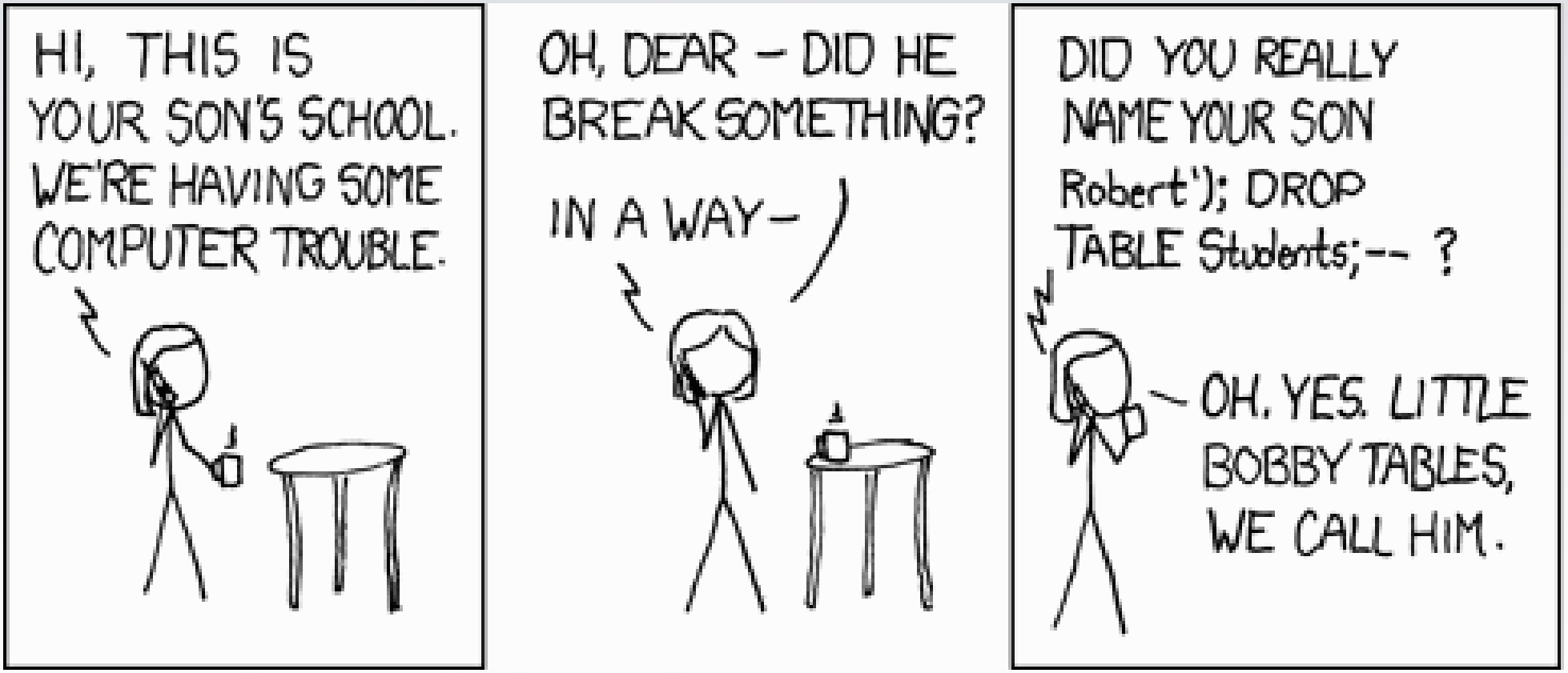
Seguridad a nivel de la aplicación: Usuarios, Roles, Permisos y Privilegios incrustados en la lógica de negocio, restricciones de acceso a los recursos, propietarios de registros/objetos, visibilidades, etc.

**Aplicación**



# ¿SQL INJECTION?

(Tópico Aparte,
Mover a SQL)




**¿Quién puede explicar la razón por la que esta tira cómica es tan (pero tan) graciosa? (al menos para mi)**

**(Cortesía de xkcd)**

```
SELECT * FROM user WHERE
name='%NAME%' AND pass='%PASS%'
```

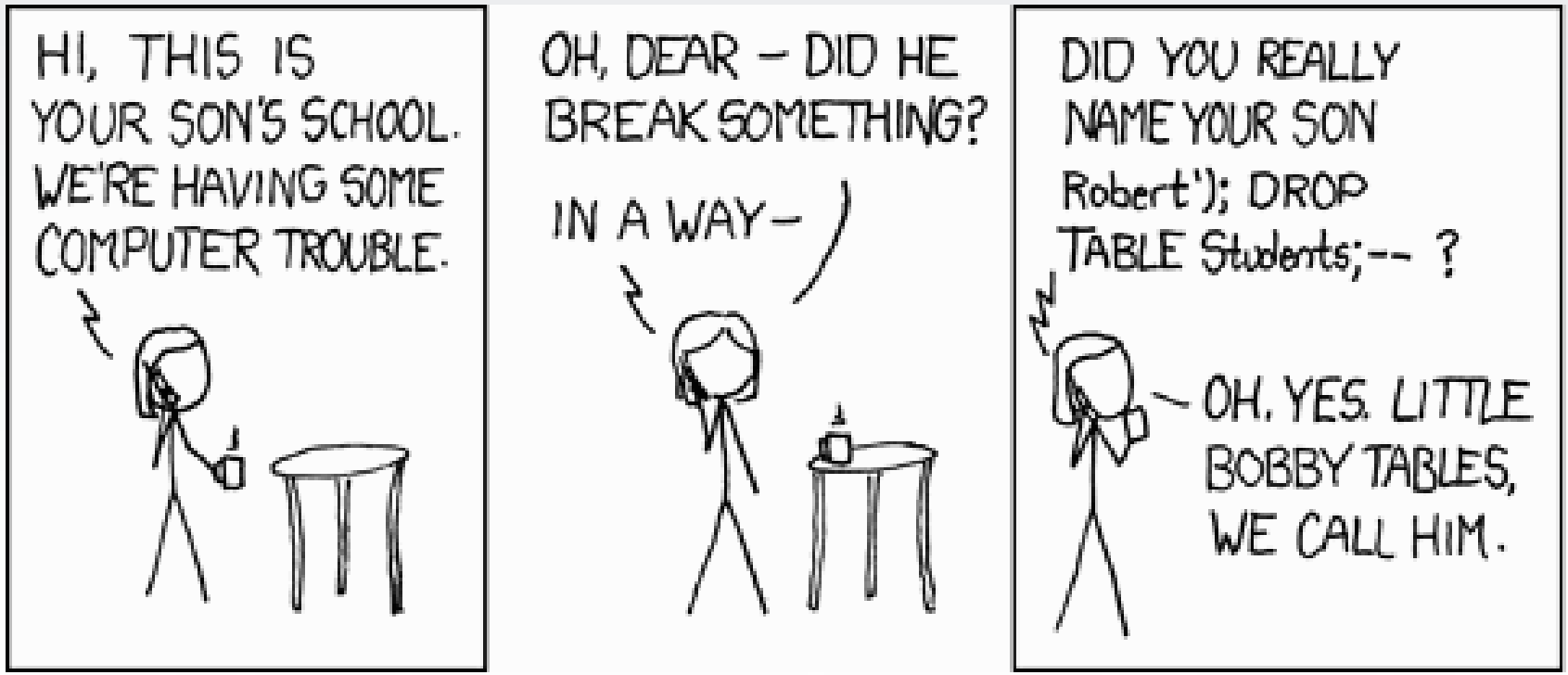
(Si retorna al menos un registro, dejamos entrar al usuario)

```
%NAME% = foo
%PASS% = whatever' OR '1'='1
```

**¿Cuál será el SQL generado?**

```
SELECT * FROM user WHERE name='foo' AND
pass='whatever' OR '1'='1'
```

**¿Cuántos registros genera la consulta anterior?**




**¿Les resulta ahora gracioso?**

# ...volviendo al tópico (Seguridad)

# Seguridad
# (A nivel del SGBD)

UNIVERSIDAD DE LOS ANDES VENEZUELA

En los SGBD, el concepto de seguridad se refiere a la ***protección de los datos*** ante ***usuarios no autorizados***, es decir, definir estrategias que permitan establecer que usuarios pueden acceder a que datos

- Tipos de seguridad en los SGBD
  - Seguridad Discrecional
  - Seguridad Obligatoria
  - Seguridad en Sistemas Estadísticos

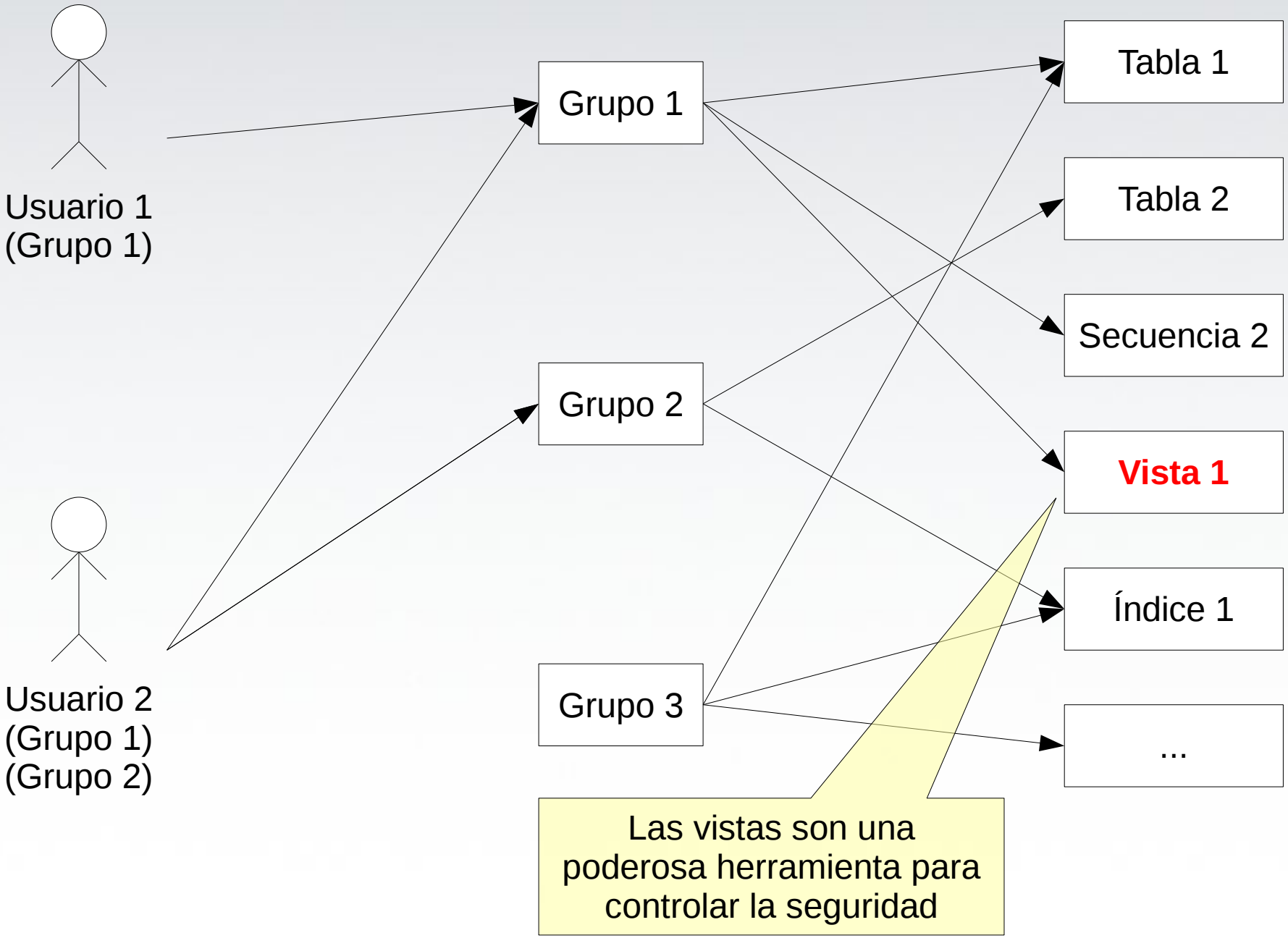
**(¿Qué creen que sea esto?)**

# Seguridad Discrecional

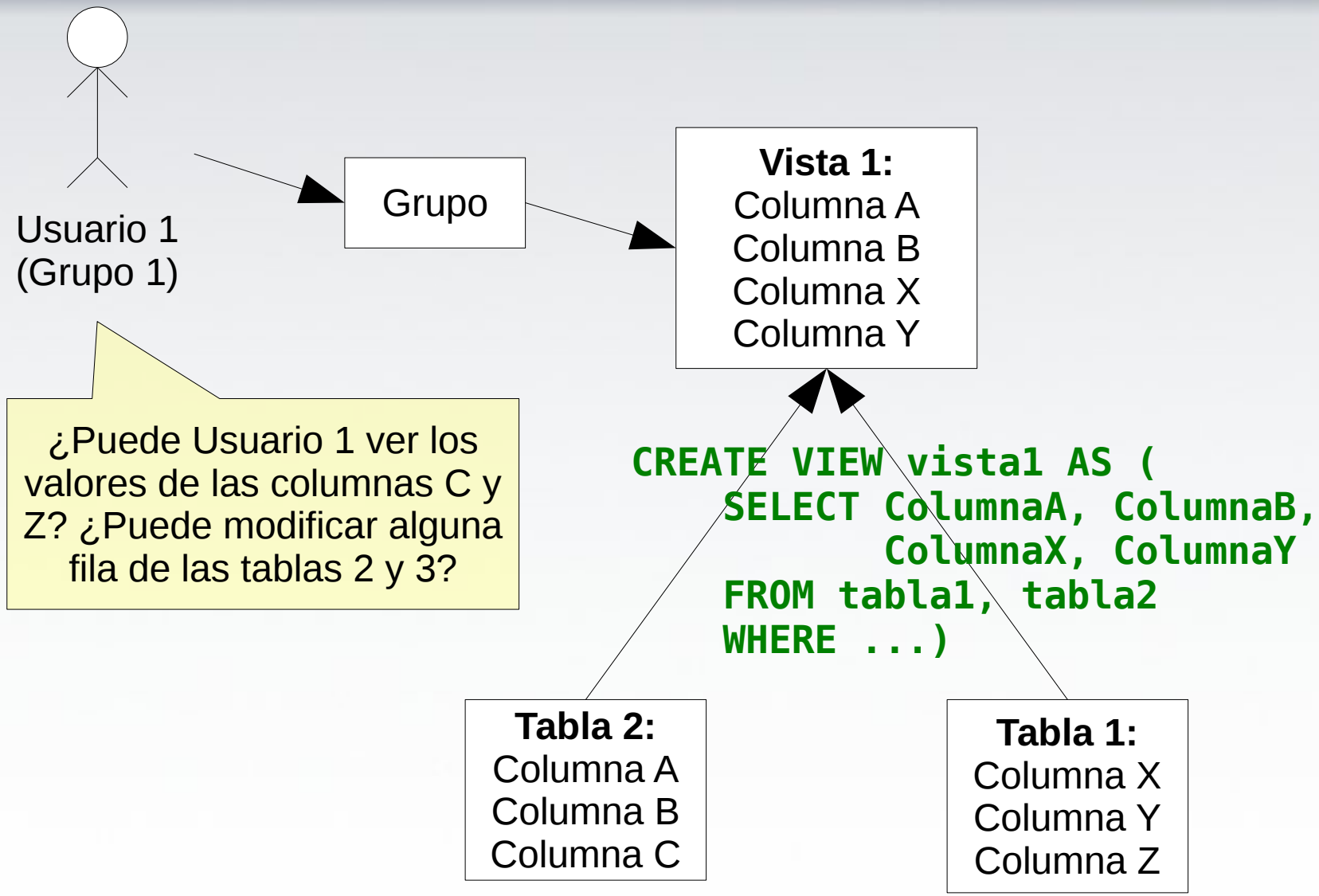
Se basa en otorgar privilegios a usuarios (o grupos de usuarios), en los que se incluye la capacidad de tener acceso tablas, registros o campos específicos con un determinado modo (para leer, insertar o actualizar)

- Autorizar al usuario X a realizar consultas en filas de la tabla A
- Autorizar al usuario X a utilizar un procedimiento almacenado B

# Seguridad Discrecional



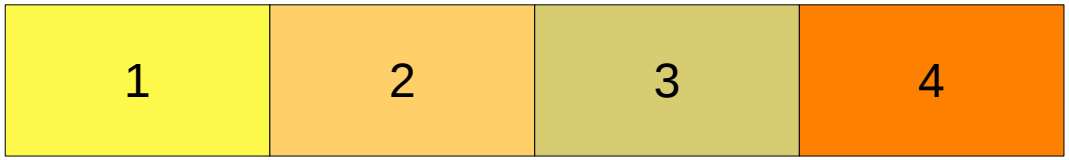
# Seguridad Discrecional




**¿Cómo evitan que Usuario 1 vea ciertas filas específicas?**

Consiste en imponer seguridad de múltiples niveles, clasificando los datos y los usuarios en varias clases (o niveles) de seguridad, de manera que los usuarios puedan acceder a los datos según tengan o no el nivel necesario para el dato que desean acceder

- Las filas (o los objetos) tienen un nivel Fi de seguridad, que solo se pueden leer si el usuario tiene un nivel Ui >= Fi de seguridad...

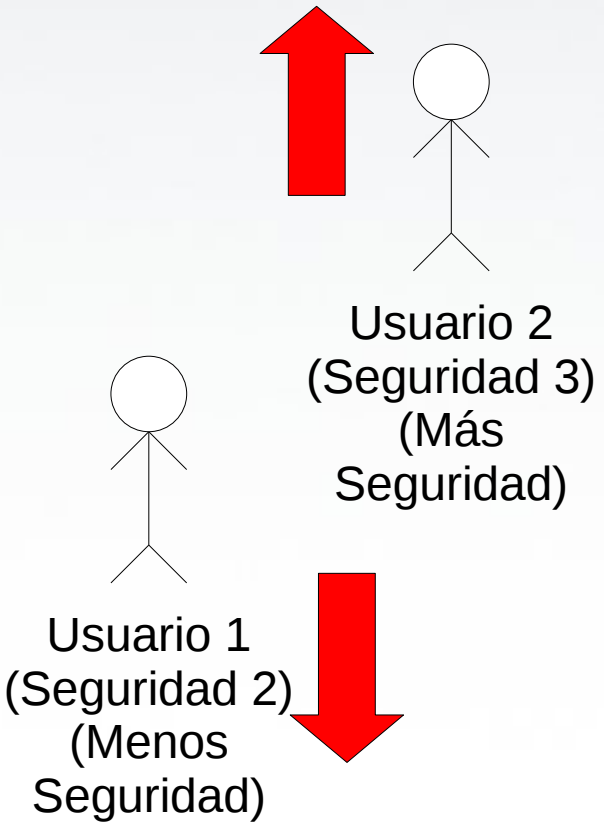


# Seguridad Obligatoria (Reglas)

UNIVERSIDAD DE LOS ANDES VENEZUELA

- El usuario X puede recuperar (leer) el objeto Y sólo si el nivel de acreditación (seguridad) de X (Ui) es mayor o igual que el nivel de clasificación de Y (Fi)
  ("propiedad de seguridad simple)
- El usuario X puede actualizar el objeto Y sólo si el nivel de acreditación de X (Ui) es **igual** al nivel de clasificación de Y (Fi) ("propiedad estrella"). **¿Ideas de por qué esto es así?**

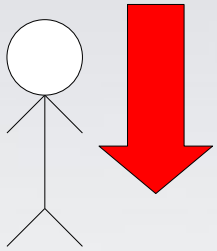
La segunda regla anterior evita que existan filtraciones de seguridad hacia abajo, o que un usuario escriba datos que luego no pueda leer... (hacia arriba)




**Empleado**

| Cédula | Nombre | Sueldo | Clasificación |
|---|---|---|---|
| 12.334.543 | Pedro Perez | 3000 | 3 |
| 14.232.650 | Juan García | 2500 | 3 |
| 15.556.345 | Diego Rojas | 4000 | 3 |
| 16.343.222 | Luis Silva | 10000 | 4 |
| 18.909.123 | Marcos Quintero | 1000 | 2 |

# Seguridad Obligatoria (Reglas)



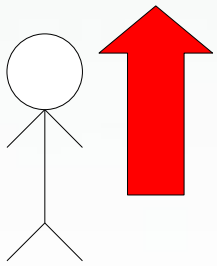
Usuario 1
(Seguridad 2)

**SELECT * FROM empleado**

**SELECT * FROM empleado WHERE clasificacion <= 2**

**Empleado**

| Cédula | Nombre | Sueldo | Clasificación |
|---|---|---|---|
| 18.909.123 | Marcos Quintero | 1000 | 2 |



Usuario 2
(Seguridad 3)

**SELECT * FROM empleado**

**SELECT * FROM empleado WHERE clasificacion <= 3**

**Empleado**

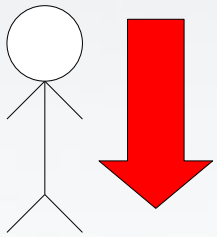
| Cédula | Nombre | Sueldo | Clasificación |
|---|---|---|---|
| 12.334.543 | Pedro Perez | 3000 | 3 |
| 14.232.650 | Juan García | 2500 | 3 |
| 15.556.345 | Diego Rojas | 4000 | 3 |
| 18.909.123 | Marcos Quintero | 1000 | 2 |

# Seguridad Obligatoria (Reglas)

UNIVERSIDAD DE LOS ANDES VENEZUELA

**INSERT INTO empleado**
**VALUES(16.343.222, 'Luis Silva', 2000)**

**INSERT INTO empleado**
**VALUES(16.343.222, 'Luis Silva', 2000, 2)**

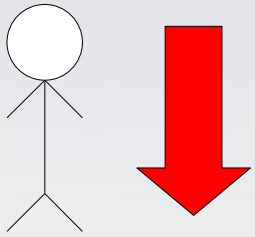


Usuario 1
(Seguridad 2)

**Empleado**

| Cédula | Nombre | Sueldo | Clasificación |
|---|---|---|---|
| 12.334.543 | Pedro Perez | 3000 | 3 |
| 14.232.650 | Juan García | 2500 | 3 |
| 15.556.345 | Diego Rojas | 4000 | 3 |
| **16.343.222** | **Luis Silva** | **2000** | **2** |
| 16.343.222 | Luis Silva | 10000 | 4 |
| 18.909.123 | Marcos Quintero | 1000 | 2 |

# Seguridad Obligatoria (Reglas)

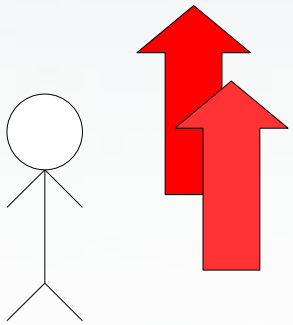


Usuario 1
(Seguridad 2)

**SELECT * FROM empleado**
**SELECT * FROM empleado WHERE clasificacion <= 2**

**Empleado**

| Cédula | Nombre | Sueldo | Clasificación |
|---|---|---|---|
| 18.909.123 | Marcos Quintero | 1000 | 2 |
| **16.343.222** | **Luis Silva** | **2000** | **2** |



Usuario 3
(Seguridad 4)

**SELECT * FROM empleado**
**SELECT * FROM empleado WHERE clasificacion <= 4**

**Empleado**

| Cédula | Nombre | Sueldo | Clasificación |
|---|---|---|---|
| 12.334.543 | Pedro Perez | 3000 | 3 |
| 14.232.650 | Juan García | 2500 | 3 |
| 15.556.345 | Diego Rojas | 4000 | 3 |
| **16.343.222** | **Luis Silva** | **2000** | **2** |
| 16.343.222 | Luis Silva | 10000 | 4 |
| 18.909.123 | Marcos Quintero | 1000 | 2 |

UNIVERSIDAD DE LOS ANDES VENEZUELA

## Oracle Label Security

http://www.oracle.com/technology/deploy/security/database-security/label-security/index.html

## SEPostgreSQL (Security Enhaced)

http://wiki.postgresql.org/wiki/SEPostgreSQL_Introduction

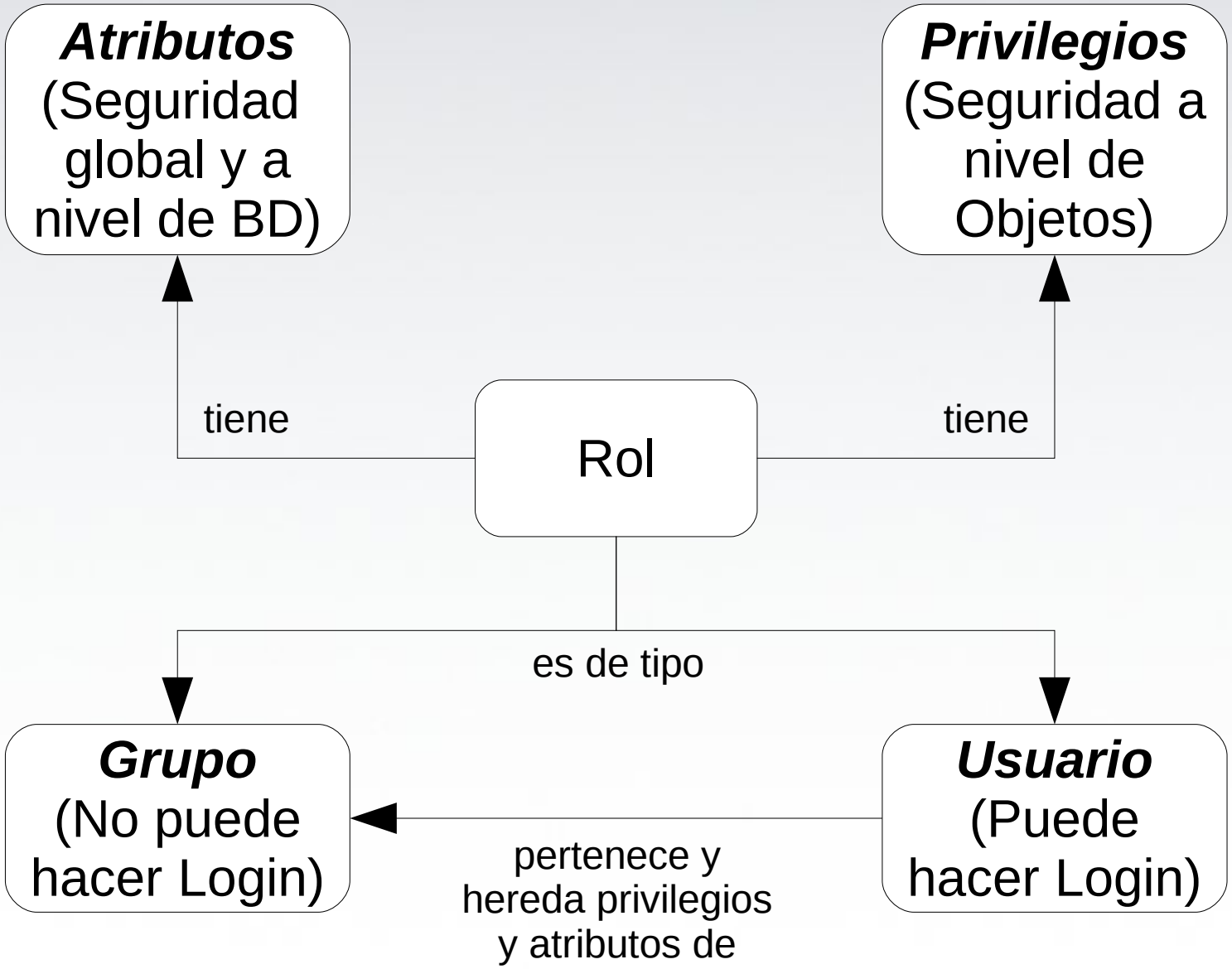
# Seguridad Discrecional (PostgreSQL)

UNIVERSIDAD DE LOS ANDES VENEZUELA

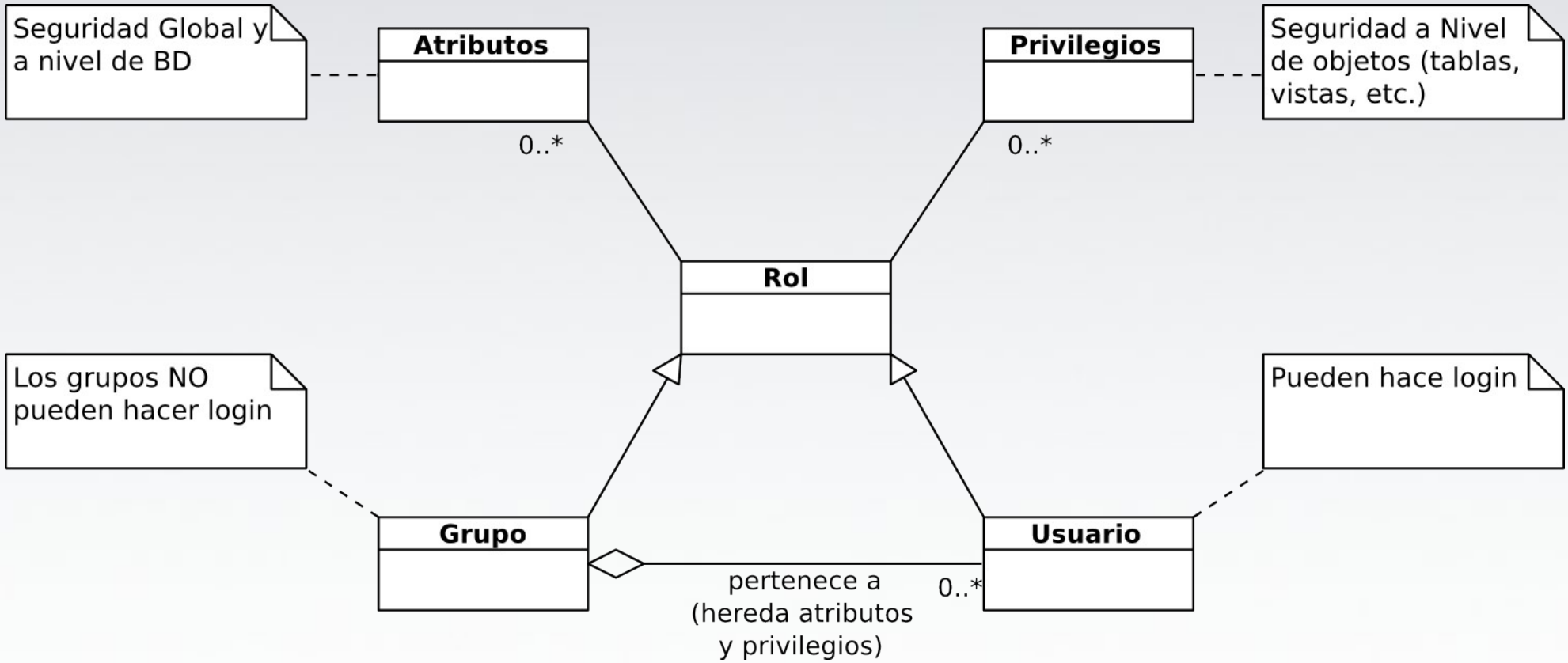
PostgreSQL manages database access permissions using the concept of roles. A role can be thought of as ***either a database user, or a group of database users***, depending on how the role is set up. ***Roles can own database objects*** (for example, tables) and can assign privileges on those objects to other roles to control who has access to which objects. Furthermore, it is possible to grant membership in a role to another role, thus allowing the member role use of privileges assigned to the role it is a member of.

***Fuente:*** http://www.postgresql.org/docs/8.4/interactive/user-manag.html

# Seguridad Discrecional (PostgreSQL)



# Seguridad Discrecional (PostgreSQL)




**Advertencia: La herencia entre Grupo, Usuario y Rol y la relación entre Grupo y Usuario tiende a crear mucha confusión en lo que respecta a la terminología...**

Los roles de la BD son “usuarios”, son globales, es decir, no existen por cada base de datos

**CREATE ROLE nombre_rol;**

**DROP ROLE nombre_rol;**

**SELECT * FROM pg_roles;**

```
rolname |super|inherit|createrole|createdb|catupdate|canlogin|connlimit|pass|...
--------+-----+-------+----------+--------+---------+--------+---------+----+---
postgres|t    |t      |t         |t       |t        |t       |       -1|****|...
foo     |f    |t      |f         |t       |f        |t       |       -1|****|...
```

Para más información sobre el resultado del select consultar el manual de PostgreSQL

# Seguridad Discrecional (PostgreSQL / Atributos)

UNIVERSIDAD DE LOS ANDES VENEZUELA

- **LOGIN**: La posibilidad de hacer login
- **SUPERUSER**: Permisos de superusuario (usar con cuidado)
- **CREATEDB**: La posibilidad de crear bases de datos
- **CREATEROLE**: La posibilidad de crear otros roles
- **PASSWORD 'string'**: Para asignar una contraseña (String es el password)

```
CREATE ROLE foo LOGIN CREATEDB PASSWORD 'foo123';

ALTER ROLE foo CREATEDB PASSWORD 'xxxyyy';
```

# Seguridad Discrecional (PostgreSQL / Sobre el password)

UNIVERSIDAD DE LOS ANDES VENEZUELA

## Archivo de Configuración de PostgreSQL /usr/local/pgsql/data/pg_hba.conf

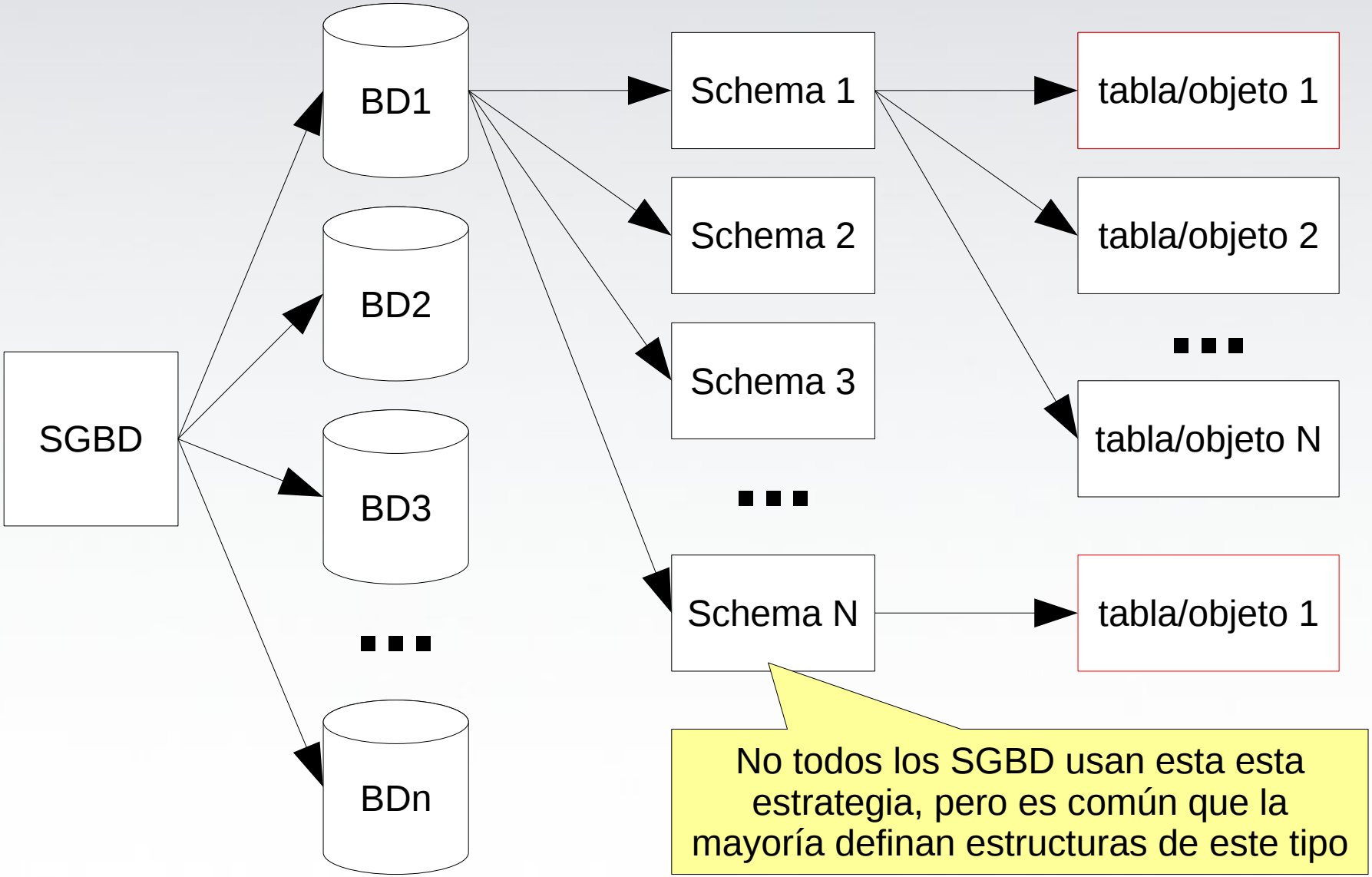
```
#TYPE      DATABASE    USER      CIDR-ADDRESS       METHOD

# "local" is for Unix domain socket connections only
#local      all         all                          trust
 local      all         all                          password

# IPv4 local connections:
#host       all         all       127.0.0.1/32       trust
 host       all         all       127.0.0.1/32       password

# IPv6 local connections:
#host       all         all       ::1/128            trust
 host       all         all       ::1/128            password
```

**Es decir, la estrategia de autentificación puede variar y ajustarse según las necesidades (No solo en PostgreSQL, sino en la mayoría de los SGBD)**

# ESQUEMAS
# SCHEMAS

(Tópico Aparte,
Mover a SQL)

# Seguridad Discrecional (PostgreSQL / Privilegios)



# Crear o Eliminar una Schema (¿Qué es un Schema?)

UNIVERSIDAD DE LOS ANDES VENEZUELA

```
test=# \dn
        List of schemas
        Name         |  Owner
---------------------+----------
 information_schema | postgres
 pg_catalog         | postgres
 pg_toast           | postgres
 pg_toast_temp_1    | postgres
 public             | postgres
(5 rows)

test=# CREATE SCHEMA some_schema_name;
CREATE SCHEMA
test=# \dn
        List of schemas
        Name         |  Owner
---------------------+----------
 information_schema | postgres
 pg_catalog         | postgres
 pg_toast           | postgres
 pg_toast_temp_1    | postgres
 public             | postgres
 some_schema_name   | postgres
(6 rows)

test=#
```

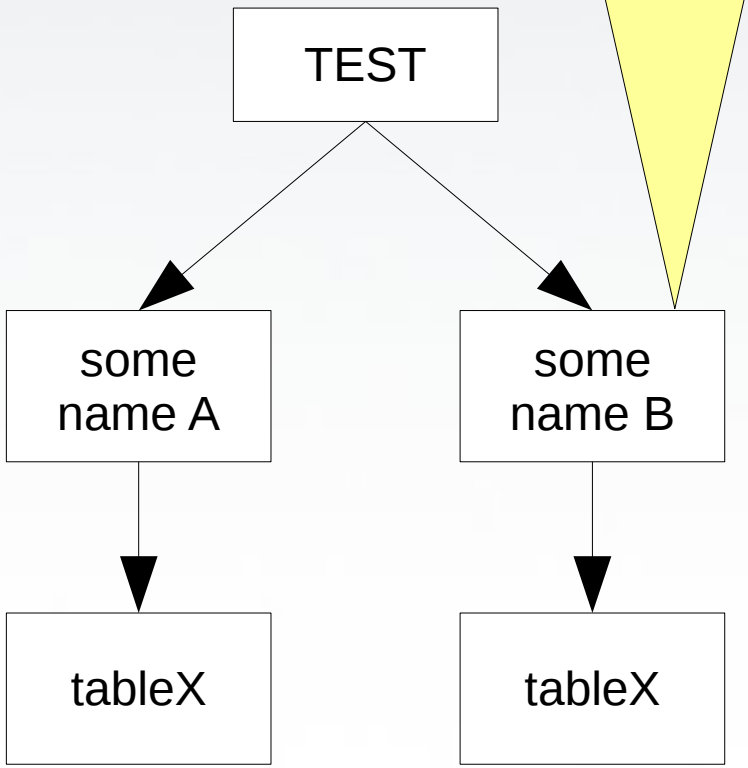
El schema es básicamente un espacio de nombre, es posible tener tablas con el mismo nombre en diferentes schemas

```
test=# DROP SCHEMA some_schema_name;
DROP SCHEMA
test=# \dn
        List of schemas
        Name         |  Owner
---------------------+----------
 information_schema | postgres
 pg_catalog         | postgres
 pg_toast           | postgres
 pg_toast_temp_1    | postgres
 public             | postgres
(5 rows)

test=#
```

# Seguridad Discrecional (PostgreSQL / Privilegios)

```
test=# CREATE SCHEMA some_name_a;
CREATE SCHEMA
test=# CREATE SCHEMA some_name_b;
CREATE SCHEMA
test=# CREATE TABLE  some_name_a.tableX
           (id INT);
CREATE TABLE
test=# CREATE TABLE  some_name_b.tableX
           (id INT);
CREATE TABLE
test=# INSERT INTO   some_name_a.tableX
           VALUES (1);
INSERT 0 1
test=# INSERT INTO   some_name_b.tableX
           VALUES (9);
INSERT 0 1
test=# SELECT * FROM some_name_a.tableX;
 id
----
  1
test=# SELECT * FROM some_name_b.tableX;
 id
----
  9
```

Los schemas ayudan en la seguridad en el sentido de que son **OBJETOS** a los que les podemos asignar privilegios y que agrupan a **OTROS OBJETOS**



...volviendo al tópico (Seguridad)

Cuando un “objeto” (tabla, secuencia, índice, etc) es creado usualmente se le asigna un dueño. Por defecto, el dueño es el rol (usuario) que crea el objeto. Es decir, para la mayor parte de los objetos, inicialmente, sólo el dueño y el superusuario pueden hacer algo con el objeto

Para permitir a otros usuarios hacer algo con esos objetos es necesario asignar privilegios, de los cuales hay de distintos tipos:
***(Lamina siguiente)***

***SELECT, INSERT, UPDATE, DELETE, REFERENCES, TRIGGER, CREATE, CONNECT, TEMPORARY, EXECUTE, USAGE***

Los privilegios se asignan con el comando GRANT:

**GRANT UPDATE ON tabla TO rol;**

y se eliminan con REVOKE:

**REVOKE ALL ON tabla FROM rol;**

Ejemplo en SQL con SELECT e INSERT...

**SELECT**

Permite hacer SELECT de cualquier columna en la tabla, vista o secuencia especificada. También permite usar COPY FROM. Para secuencias este privilegio permite usar la función currval.

**INSERT**

Permite insertar nuevas filas en la tabla especificada, también permite la operación COPY TO sobre la tabla en cuestión.

## UPDATE

Permite actualizar cualquier columna de la tabla especificada. SELECT ... FOR UPDATE y SELECT ... FOR SHARE también requieren este privilegio (Además del privilegio SELECT). Para secuencias, este privilegio permite el uso de la función nextval y setval.

## DELETE

Permite eliminar filas de la tabla en cuestión.

**REFERENCES**

Para crear claves foráneas es necesario tener este privilegio tanto en la tabla que hace la referencia como en la tabla referenciada.

**TRIGGER**

Permite crear gatillos en la tabla especificada

## CREATE

Si se asocia a **bases de datos**, permite crear **esquemas** dentro de la base de datos.

Si se asocia a **esquemas** permite crear **objetos** (tablas, vistas, etc) dentro del esquema. Para cambiar el nombre de un objeto es necesario tener este privilegio.

**CONNECT**

Permite a los usuarios conectarse a la base de datos especificada.

**TEMPORARY**
**TEMP**

Permite la creación de tablas temporales en la base de datos especificada.

**EXECUTE**

Permite el uso de la función especificada
(Sólo aplica a funciones)

**USAGE**

Para lenguajes de procedimientos permite el uso del lenguaje especificado (para crear funciones en el lenguaje especificado)

Para **esquemas** permite el acceso a los objetos contenidos en el esquema (asumiendo que los privilegios de los objetos particulares se cumplen)

Para secuencias, permite el uso de la función currval y nextval sobre la secuencia especificada.

**ALL PRIVILEGES**

Da todos los privilegios disponibles. La palabra PRIVILEGES es opcional en PostgreSQL pero su uso es obligatorio en el estándar SQL.

# Un ejemplo de la clase de problemas que se pueden encontrar...

Native PostgreSQL restricts DB users with ACLs on views. This means that access paths to a certain table cannot be described clearly, because multiple views to a table may be defined. Moreover, an ACL on a table is not evaluated when accessed from a view. Thus, a DB user who has the authority to define a view can access any information in the database.

UNIVERSIDAD DE LOS ANDES VENEZUELA

# ¡Gracias!

# Seguridad

UNIVERSIDAD DE LOS ANDES VENEZUELA

- Añadir
  - Seguridad en sistemas estadísticos (O dejarlo como tarea)