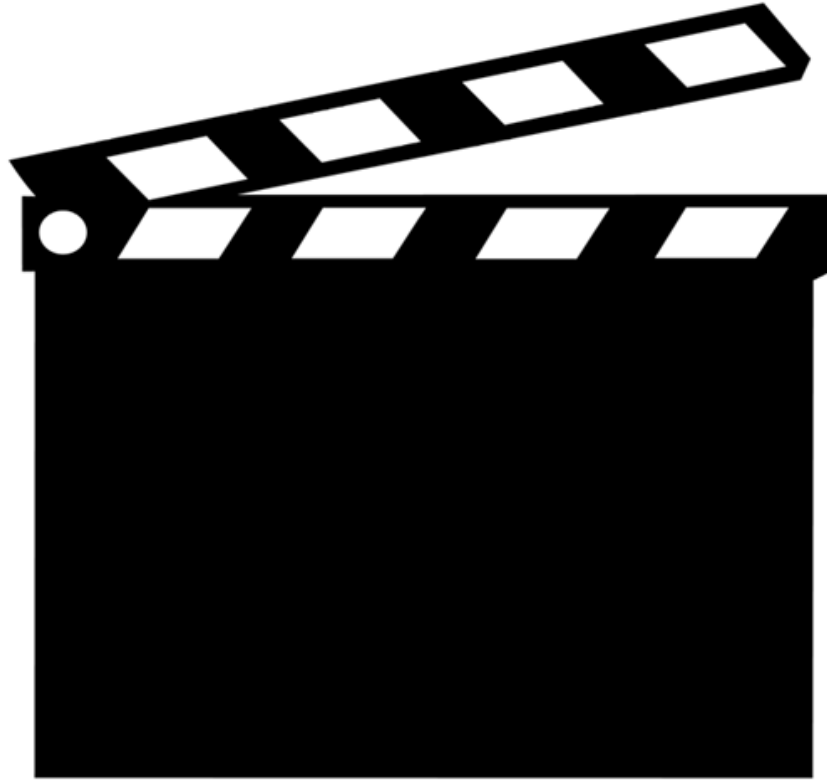


C018SA-W4-S2



SEMAINE 4 : Contrôle d'Accès

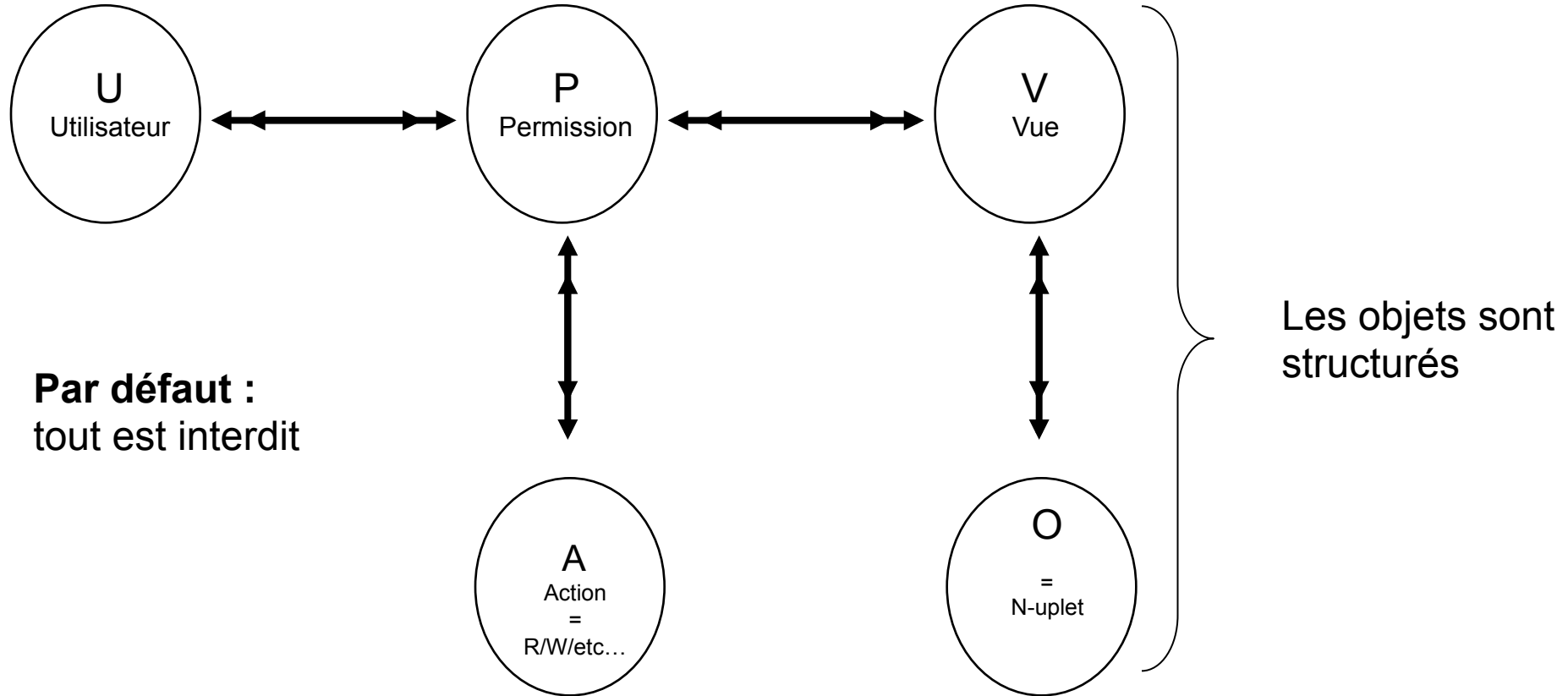
1. Introduction

**2. Modèle de contrôle d'accès
discrétionnaire (DAC)**

3. Modèle de contrôle d'accès basé sur les
rôles (RBAC)

4. Modèle de contrôle d'accès obligatoire
(MAC)

Le modèle DAC en SQL (92)



Le modèle DAC

- **DAC** = Discretionary Access Control ou Contrôle d'Accès Discrétionnaire (i.e. à la discrétion du **propriétaire** de la donnée)
- **Principes**
 - Le créateur d'un objet est le propriétaire de cet objet
 - Le propriétaire fixe la politique de contrôle d'accès sur cet objet
 - Les sujets reçoivent des permissions (droits) pour réaliser des actions sur cet objet
 - Les sujets ont l'autorisation de transférer certaines permissions à d'autres sujets
- **Modèle décentralisé**
 - *Avantage* : Souplesse
 - *Inconvénient* : Difficulté à administrer

Le modèle DAC

- Les droits sont représentés sous forme de matrice
- Remplie par :
 - Capacités (ligne, Capability List)
 - Autorisations (colonne, Access Control List)
- La matrice peut être très grande...

	RH	Paie	Salaire	Heures Sup	...
Alice	R/W				
Bob	R/W				
Charlie		R/W	R/W		
...					

Expression du DAC en SQL : Commande GRANT

```
GRANT <liste privileges>  
ON <table ou vue>  
TO <liste utilisateurs>  
[ WITH GRANT OPTION ] ;
```

- **Privilèges** : SELECT, INSERT, UPDATE, UPDATE(nom_colonne), DELETE
- WITH GRANT OPTION est optionnel et permet la délégation de droits

Expression du DAC en SQL : Commande REVOKE

```
REVOKE [GRANT OPTION FOR] <liste privileges>  
ON <table ou vue>  
FROM <liste utilisateurs>  
[ RESTRICT / CASCADE ] ;
```

- **RESTRICT / CASCADE** : permet de gérer la délégation de droit. Supposons que D donne le droit de délégation du privilège p à C, puis C le donne à B.
 - **CASCADE** : si D révoque le droit à C alors B le perd
 - **RESTRICT** : si D révoque le droit à C alors B le garde

Contrôle d'accès basé sur les vues : Fonctionnement

- Définition de Vues
- Droits donnés sur les vues
- Interrogation des vues et non des tables « primaires » par les utilisateurs
 - Vue DRH :

```
CREATE VIEW DRH AS  
SELECT ID-E, NOM, ADRESSE, VILLE, TEL  
FROM EMP
```

- Vue DAF :

```
CREATE VIEW DAF AS  
SELECT ID-E, NOM, SALAIRE  
FROM EMP
```

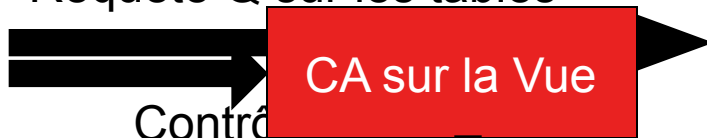
Il suffit de contrôler l'accès aux vues

Fonctionnement

Alice, RH



Requête Q sur les tables



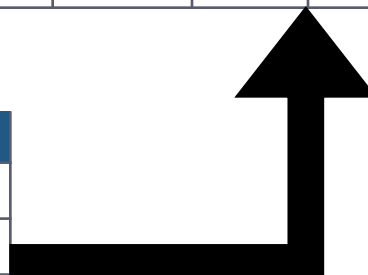
Id-E	Nom	Tel	Adresse	Ville	Salaire
1	Alice	1 2345	Rue Monge	Paris	2300
2	Bob	1 2346	Av EU	Versailles	1200
3	Charlie	1 2347	PI Gordaine	Bourges	3800
4	Diane	1 2348	Av d' Italie	Paris	1600

Requête Q' sur les vues



Id-E	Nom	Tel	Adresse	Ville
1	Alice	1 2345	Rue Monge	Paris
2	Bob	1 2346	Av EU	Versailles
3	Charlie	1 2347	PI Gordaine	Bourges
4	Diane	1 2348	Av d' Italie	Paris

Requête Q'' sur les tables



Exemples 1/2

- Alice et Bob ont le droit de lire et modifier toutes les informations RH des dossiers admin tous les employés

```
GRANT SELECT, UPDATE ON DRH TO ALICE, BOB
```

- Charlie et Diane ont le droit de lire toutes les informations Salaire des employés

```
GRANT SELECT, UPDATE ON DAF TO CHARLIE, DIANE
```

Exemples 2/2

- Eric et Fiona ont le droit de lire les informations de salaire de leur propre dossier
→ On pourrait créer une vue par utilisateur avec juste ses données
→ On peut aussi utiliser un opérateur prédéfini SQL : `CURRENT_USER`

```
CREATE VIEW PERSO AS  
SELECT SALAIRE FROM EMP WHERE ID-E = CURRENT_USER;
```

```
GRANT SELECT ON PERSO TO ERIC, FIONA
```

Expressivité

- Forte, mais complexe

« En l'absence de Diane, permettre à Charlie de gérer ses dossiers »

« Permettre l'accès aux données uniquement aux heures ouvrables »

1. Créer des tables pour gérer les *meta-données* ou utiliser des meta-données système
2. Créer des requêtes utilisant ces tables
3. Donner les droits