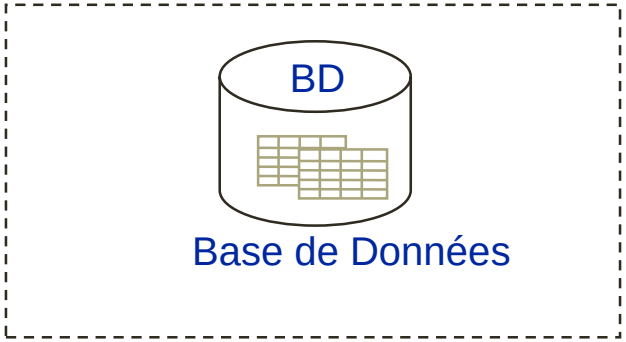
SGBD 3
COURS N° 4

Les Vues relationnelles



Jérôme FESSY

SQL : Un Langage standardisé

Programming Language

PL
PROCEDURE/FUNCTION / TRIGGER

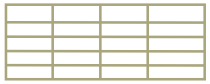
Langage de Manipulation des Données

SQL LMD
SELECT/INSERT/UPDATE/DELETE

Langage de Définition des Données

SQL LDD
CREATE/DROP/ALTER

INDEX
VUE
TRIGGER
PROCEDURE STOCKEE
SEQUENCE
TABLE



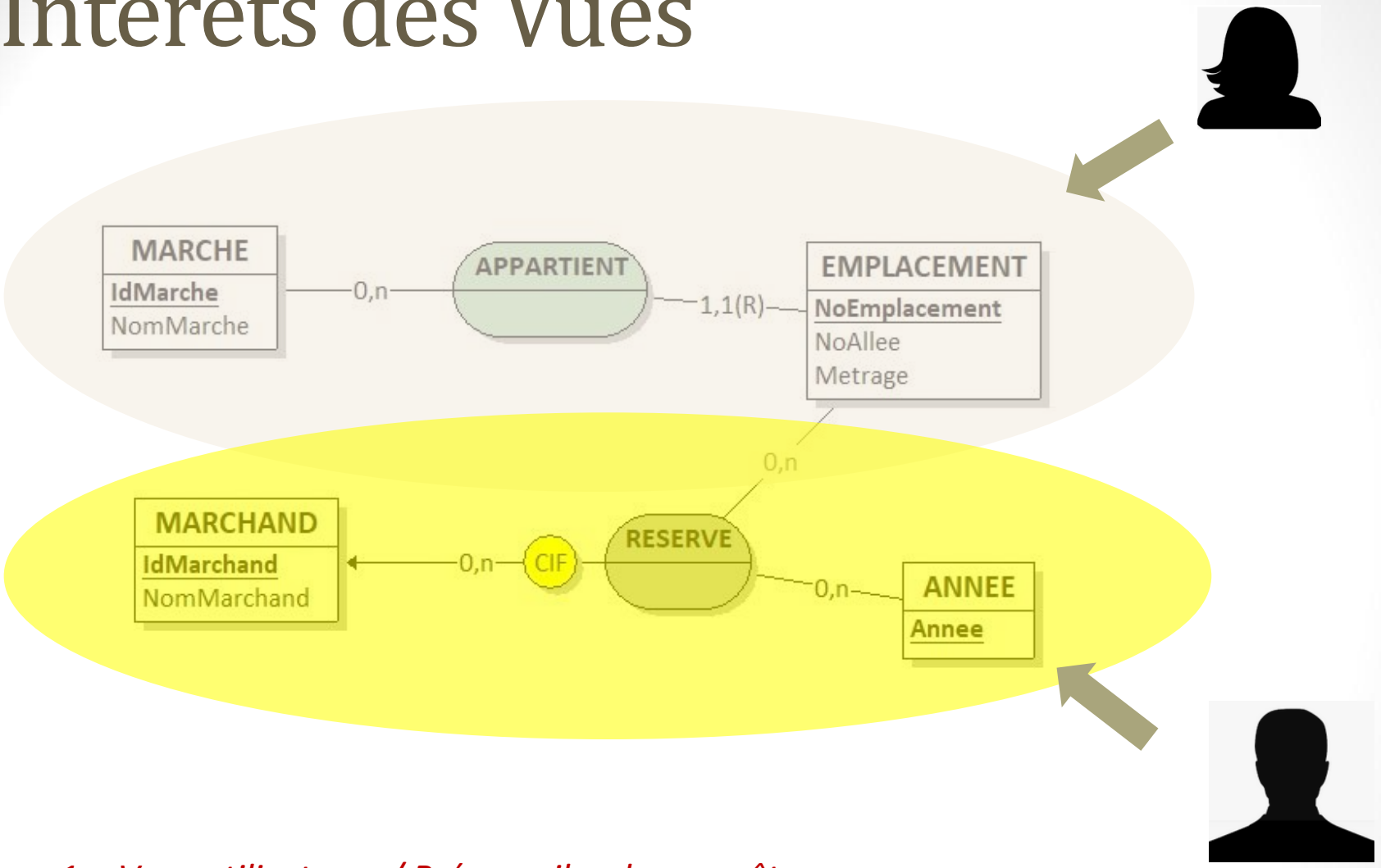
Données

Base de Données

Norme SQL

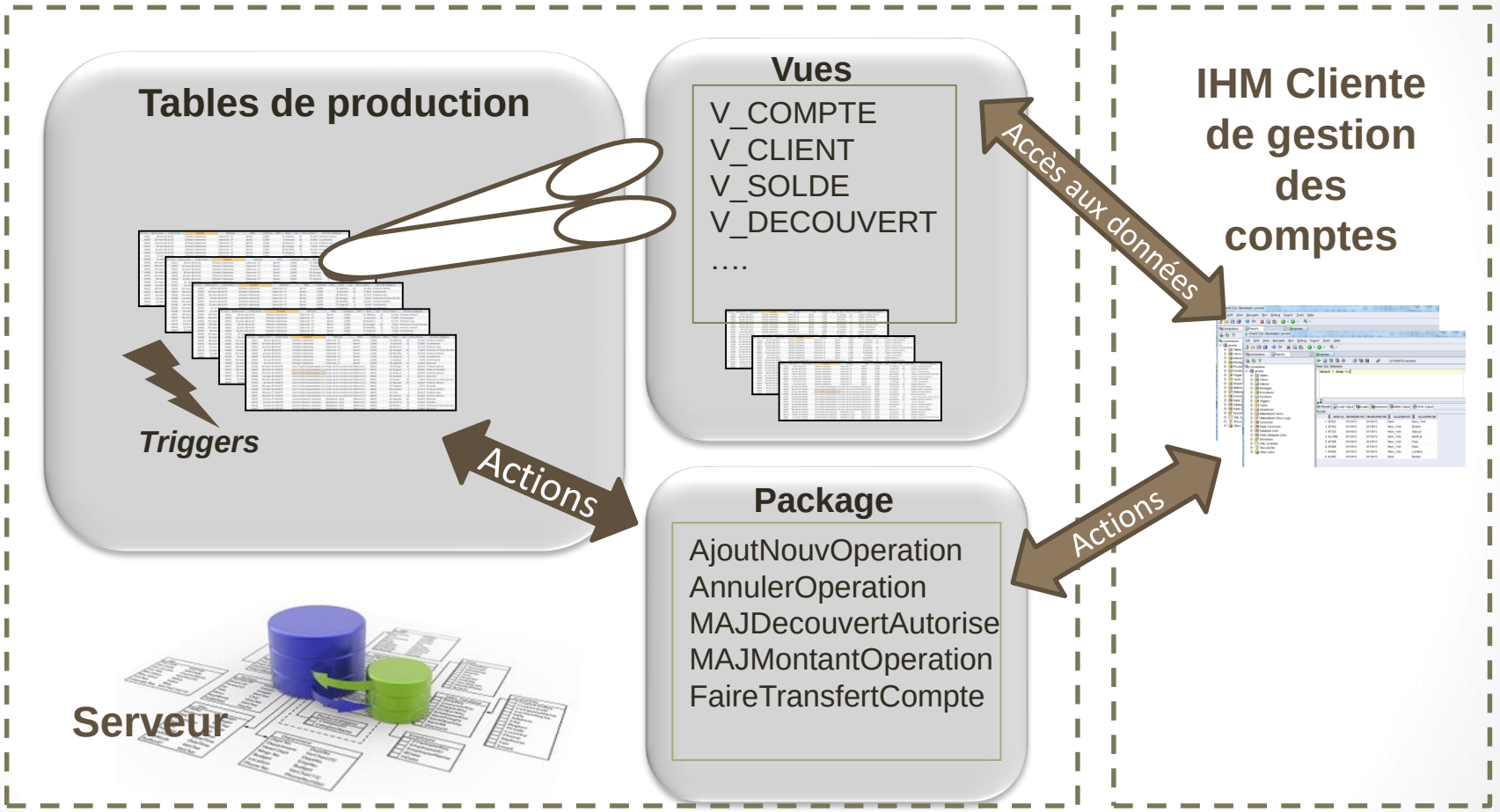
SQL1 1986 → SQL2 1992 → SQL3 2003 → SQL4

Intérêts des Vues



1. *Vues utilisateurs / Précompiler des requêtes*
2. *Restreindre/ Adapter l'accès aux données des tables*
3. *S'abstraire de la production*

Une VUE est une pseudo-table (requête nommée)

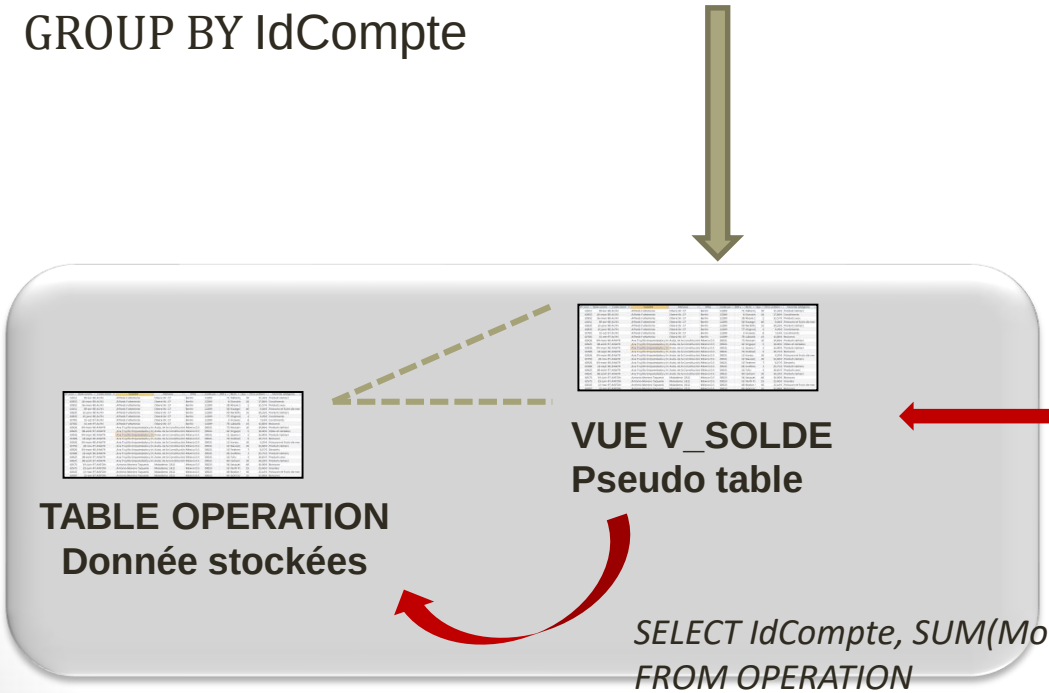


Jérôme FESSY

Usage des VUES relationnelles

1. Précompiler des vues utilisateurs / Requêtes calculées

```
CREATE OR REPLACE VIEW V_SOLDE (IdCpt, Montant)
AS
SELECT IdCompte, SUM(MontantOperation)
FROM OPERATION
GROUP BY IdCompte
```



REQUETE :

```
SELECT *
FROM V_SOLDE
WHERE IdCpt=1;
```

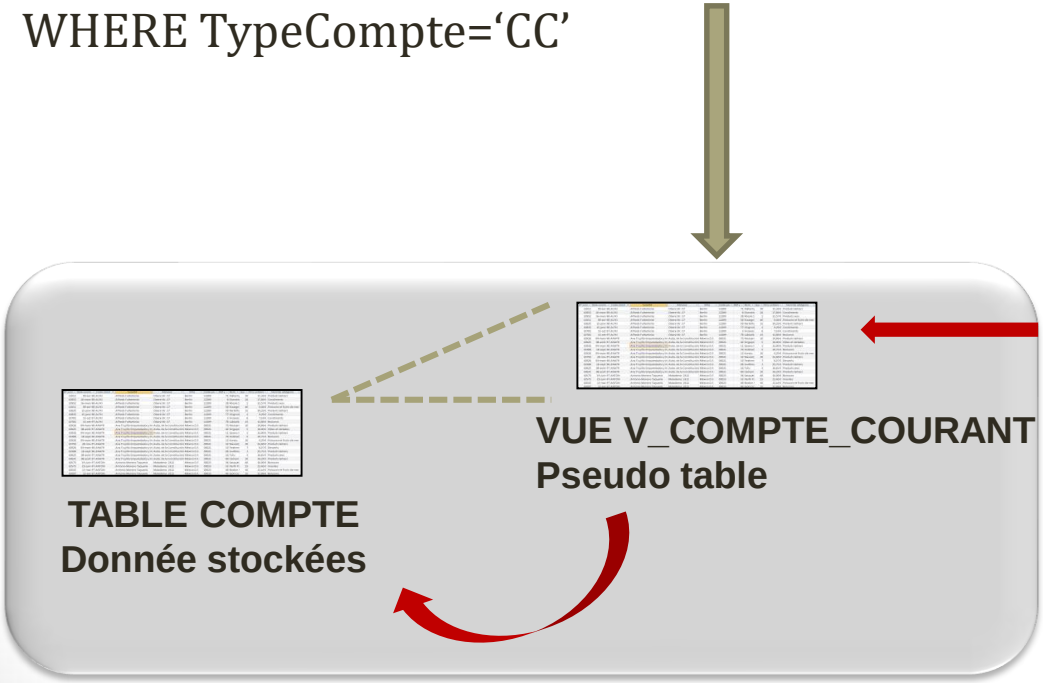
SELECT IdCompte, SUM(MontantOperation)
FROM OPERATION
WHERE IdCpt=1
GROUP BY IdCompte

Jérôme FESSY

Usage des VUES relationnelles

2. Restreindre et adapter l'accès aux données

CREATE OR REPLACE VIEW V_COMPTE_COURANT (IdCpt, Montant)
AS
SELECT IdCompte, LibelleCompte
FROM COMPTE
WHERE TypeCompte='CC'



REQUETE :

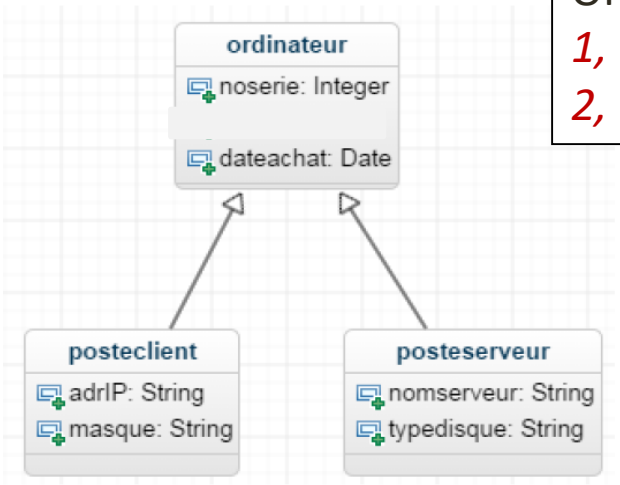
SELECT *
FROM V_COMPTE_COURANT

L'application ne dispose que de l'accès aux comptes courants

Jérôme FESSY

USAGE DES VUES

3. S'abstraire de la production (1/4)



ORDINATEURS
1, '12/11/2021', 'JUPITER', 'SSD'
2, '13/11/2021', '1:12:34:555','0:0:0:92'

2 solutions pour implanter cet héritage en relationnel

Décomposition ascendante

SOLUTION 2 SCHEMA RELATIONNELLE :

ORDINATEUR (NoSerie, Dateachat, TypeOrdi
AdrIp, Masque, NomServeur, TypeDisque)

Décomposition descendante

SOLUTION 1 SCHEMA RELATIONNELLE :

POSTECLIENT (NoSerie, Dateachat, AdrIp, Masque)
POSTESERVEUR (NoSerie, Dateachat, NomServeur, TypeDisque)

Jérôme FESSY

USAGE DES VUES

S'abstraire de la production (2/4)

ORDINATEURS
1, '12/11/2021', 'JUPITER', 'SSD'
2, '13/11/2021', '1:12:34:555','0:0:0:92'

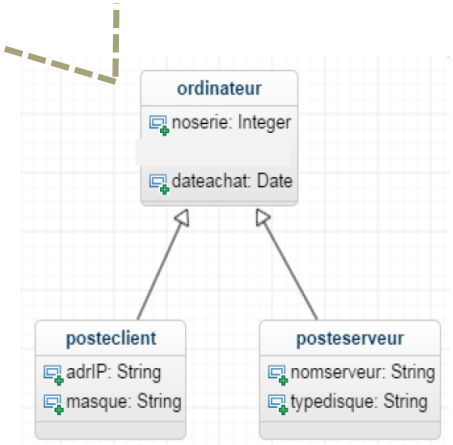
CLIENT

SERVEUR

Le schéma d'accès aux ordinateurs est le même quelque soit l'implantation choisie

V_ORDINATEUR
(NoSerie, DateAchat, TypeOrdi, AdrIp, Masque, NomServeur,TypeDisque)

Tables
Décomposition ascendante
Décomposition descendante



Jérôme FESSY

USAGE DES VUES

S'abstraire de la production (3/4)

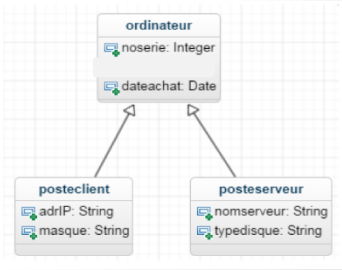
ORDINATEURS
1, '12/11/2019', 'JUPITER', 'SSD'
2, '13/11/2019', '1:12:34:555','0:0:0:92'

V_ORDINATEUR
(NoSerie, DateAchat, TypeOrdi, AdrIp, Masque, NomServeur,TypeDisque)

```
CREATE VUE V_ORDINATEUR
AS
SELECT *
FROM ORDINATEUR
```

Tables issues de la décomposition ascendante

ORDINATEUR (NoSerie, Dateachat, TypeOrdi AdrIp, Masque, NomServeur, TypeDisque)



Jérôme FESSY

USAGE DES VUES

S'abstraire de la production (4/4)

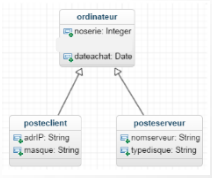
ORDINATEURS
1, '12/11/2021', 'JUPITER', 'SSD'
2, '13/11/2021', '1:12:34:555','0:0:0:92'

V_ORDINATEUR
(NoSerie, DateAchat, TypeOrdi, AdrIp, Masque, NomServeur,TypeDisque

```
CREATE VUE V_ORDINATEUR
AS
SELECT Noserie, DateAchat, 'CLIENT', NULL,NULL, AdrIP
FROM POSTECLIENT
UNION
SELECT Noserie, DateAchat, 'SERVEUR', masque,nomserveur, NULL, NULL
FROM POSTESERVEUR
```

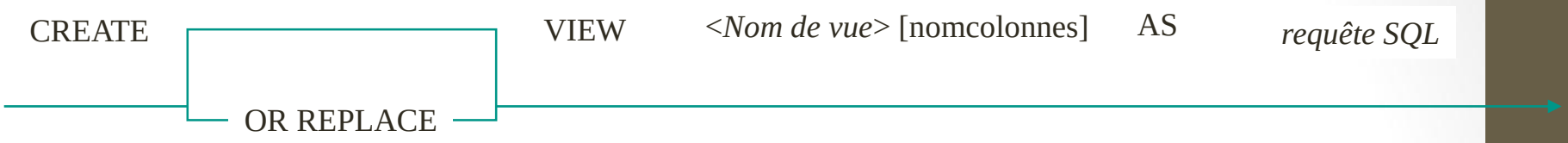
Tables issues de la décomposition descendante

POSTECLIENT (NoSerie, Dateachat, AdrIp, Masque)
POSTESERVEUR (NoSerie, Dateachat, NomServeur, TypeDisque)

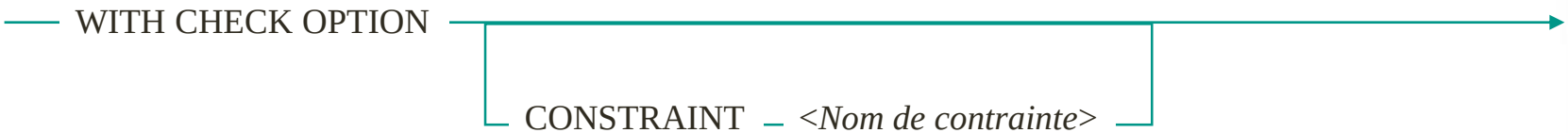


Jérôme FESSY

SYNTAXE SQL : Création d'une vue



Contrainte



Permet d'assurer que les données insérées à travers la vue sont conformes à la définition de la vue

Exemple :

```
CREATE OR REPLACE VIEW V_SOLDE (IdCpt, Solde)
AS
SELECT IdCompte, SUM(MontantOperation)
FROM OPERATION
GROUP BY IdCompte
```

Jérôme FESSY

Synthèse : vues

- Une vue ne dispose d'aucun espace de stockage
  - C'est tout simplement une requête stockée et nommée dans la base
  - Une requête sur une vue est transformée dans beaucoup de cas afin d'y intégrer la requête de la vue

- Les vues peuvent être utilisées même avec les instructions de MAJ (UPDATE, INSERT et DELETE)
  - Dans ce cas, l'opération s'effectue sur la table sous-jacente
  - Une opération de mise à jour sur une vue peut être ambiguë si la vue a une définition complexe
    - Le SGBD retourne une erreur dans ce cas
      - Sauf si un trigger INSTEAD OF a été défini sur la vue

Mises à jour au travers les vues (1/2)

INSERT INTO V_ORDINATEUR (NoSerie, DateAchat, TypeO, NomServeur,TypeDisque
VALUES (1, '12/11/2021', 'SERVEUR', 'JUPITER', 'SSD')

V_ORDINATEUR
(NoSerie, DateAchat, TypeOrdi, AdrIp, Masque, NomServeur,TypeDisque

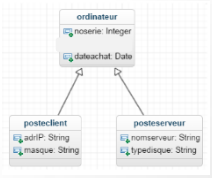
?

INSERT

Trigger INSTEAD OF
Pour programmer l'insertion
Depuis la vue dans les tables
de production

Tables issues de la décomposition descendante

POSTECLIENT (NoSerie, Dateachat, AdrIp, Masque)
POSTESERVEUR (NoSerie, Dateachat, NomServeur, TypeDisque)



Jérôme FESSY

Mises à jour au travers les vues (2/2)

Un trigger INSTEAD OF :

- **S'applique sur une vue**
- **Permet de programmer l'action à réaliser sur les tables de production lors de la mise à jour au travers de la vue**

```
CREATE OR REPLACE TRIGGER T_INSERT_ORDI
INSTEAD OF INSERT ON V_ORDINATEUR

FOR EACH ROW
BEGIN

IF :NEW.typeOrdi = 'CLIENT' THEN
INSERT INTO POSTECLIENT (NoSerie, DateAchat, AdrIp, Masque)
        VALUES (:NEW.noserie, :NEW.dateachat, :NEW.AdrIp, :NEW.Masque);
ELSE
INSERT INTO POSTESERVEUR (NoSerie, DateAchat, NomServeur, TypeDisque)
        VALUES (:NEW.noserie, :NEW.dateachat, :NEW. NomServeur, :NEW. TypeDisque);
END IF;
END;
/
```

Jérôme FESSY