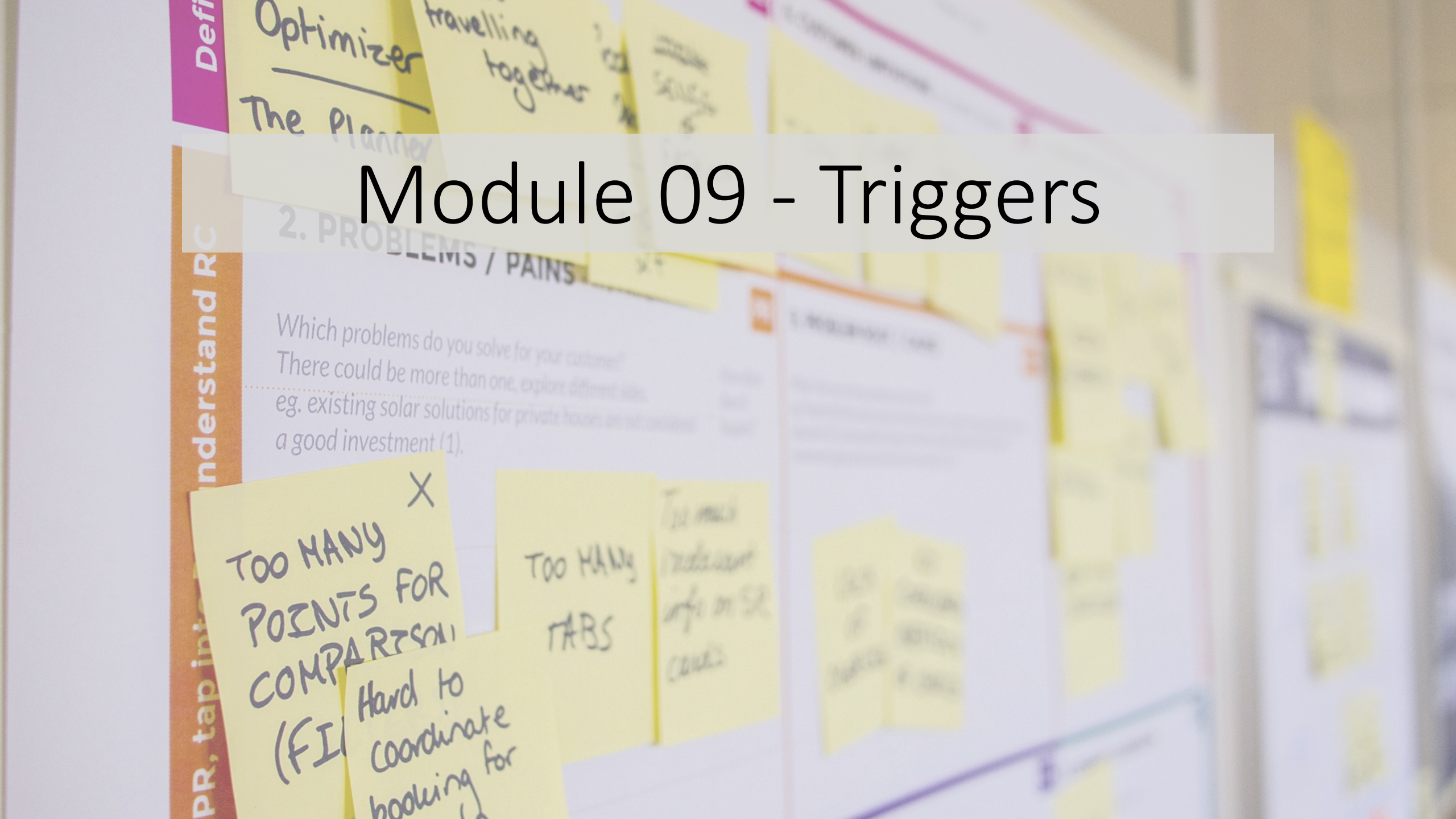

Module 09 - Triggers

Objectifs

- Définitions des triggers
- Cas d’utilisation

Triggers – Définition

- Un **trigger** (**déclencheur**) est une **procédure stockée sans paramètre** qui est **exécutée** lorsqu'un **événement spécifique** est **détecté**
- Les événements qui nous intéressent dans le cours sont les événements du DML sur une table : **insertion**, **mise à jour** et **suppression**
- Les triggers sont principalement utilisés dans deux cas :
  - Pour renforcer des règles de cohérence de données suivant les règles d'affaire (ie les validations de type CHECK ne sont pas suffisantes) : interdire le remboursement d'un montant non perçu sur l'annulation d'une commande.
  - Pour automatiser la mise à jour ou la création d'enregistrement à partir d'une action sur une table (exemple : historisation)

Triggers – Définition

- Dépendamment du moteur de base de données, on trouvera la possibilité de déclarer des triggers qui s'exécutent « **avant** » (**BEFORE**) ou « **après** » (**AFTER**) l'exécution de l'instruction de manipulation
- Les triggers de type **BEFORE** sont généralement utilisés pour modifier les valeurs de colonnes avant que l'opération d'insertions ou de mises à jour
- Les triggers de type **AFTER** sont généralement utilisés pour mettre à jour d'autres tables à partir des modifications effectuées par l'opération
- Les triggers de type **AFTER** sont exécutés après les validations de type `**CHECK**`
- Les triggers ne peuvent pas être appelés manuellement

Triggers – Exemples

- Maintenir à jour les inventaires de produits lors des ventes, modifications des ventes, annulation de ventes
- Limiter le nombre de livres empruntés dans une bibliothèque par un type de client en particulier
- Limiter le montant d'une hausse salariale
- Calcul d'une ristourne accordée au client lors d'une vente, en se basant sur ses achats antérieurs
- Conserver l'historique des changements dans une base de données
- Maintenir à jour certaines statistiques précises
- …

Triggers – SQL Server

- **SQL Server ne permet pas de définir des triggers qui s'exécutent avant l'action.** Il propose un autre type de trigger qui va agir comme une substitution à l'action initiale. (**INSTEAD OF**)
- Dans le cas des triggers de type « INSTEAD OF », il n'y a pas de récursivité en cas d'insertion ou de mise à jour de la table sur laquelle il s'applique. Ce type de triggers permet de modifier des valeurs de colonnes. Il est très utilisé dans le remplacement d'opérations appliquées sur des vues qui nécessitent d'agir sur plusieurs tables

Triggers – SQL Server

- SQL Server appelle **qu'une seule fois les triggers** enregistrés **quelque soit le nombre d'enregistrements touchés**
- Pour chaque opération d'inserts, de mises à jour et de suppressions, une transaction implicite est créée : vous pouvez l'annuler dans le cas des triggers **AFTER** (donc l'opération n'est pas exécutée)
- Dans le code du trigger, vous avez accès à deux « tables » : inserted qui contient la nouvelle version des enregistrements touchés et deleted qui contient la version remplacée :

	inserted	deleted
INSERT	Les nouveaux enregistrements	vide
UPDATE	Nouvelles valeurs	Anciennes valeurs
DELETE	vide	Enregistrements supprimés

Trigger – Syntaxe

- Les noms de triggers doivent débuter par « TR_ »

```
CREATE [ OR ALTER ] TRIGGER [ schema_name . ]trigger_name
ON { table | view }
[ WITH <dml_trigger_option> [ ,...n ] ]
{ FOR | AFTER | INSTEAD OF }
{ [ INSERT ] [ , ] [ UPDATE ] [ , ] [ DELETE ] }
[ WITH APPEND ]
[ NOT FOR REPLICATION ]
AS { sql_statement  [ ; ] [ ,...n ] | EXTERNAL NAME <method specifier [ ; ] > }

<dml_trigger_option> ::=
    [ ENCRYPTION ]
    [ EXECUTE AS Clause ]

<method_specifier> ::=
    assembly_name.class_name.method_name
```

https://learn.microsoft.com/en-us/sql/t-sql/statements/create-trigger-transact-sql

Triggers AFTER – Exemple de validations

Validation du texte du statut de la commande (La validation pourrait être réalisée par un CHECK ici – Démo –)

```
CREATE OR ALTER TRIGGER TR_Commande_ValidationsStatus ON commande
AFTER INSERT, UPDATE
AS
BEGIN
  IF EXISTS (SELECT 1 FROM inserted WHERE statut NOT IN ('Acceptée', 'Payée', 'Expédiée', 'En attente'))
  BEGIN
    ROLLBACK TRANSACTION;
    RAISERROR ('Le statut de la commande doit être une valeur conforme', 16, 1);
    -- RAISERROR si severité comprise entre 11 et 19, le moteur cherche un CATCH (Voir documentation)
  END;
END;
```

```
INSERT INTO commande(statut) VALUES ('abc');
```

=> Erreur, l'insertion est annulée

https://learn.microsoft.com/en-us/sql/t-sql/language-elements/raiserror-transact-sql

Triggers INSTEAD OF – Remplacement de valeur

Remplacement d'un code de statut par une valeur plus textuelle

```
CREATE OR ALTER TRIGGER
TR_Commande_ModifieStatus ON commande
INSTEAD OF INSERT
AS
BEGIN
	INSERT INTO commande(statut)
	SELECT CASE statut
		WHEN 'a' THEN 'Acceptée'
		WHEN 'p' THEN 'Payée'
		WHEN 'e' THEN 'Expédiée'
		ELSE statut
	END
	FROM inserted;
END;
```

```
INSERT INTO commande(statut)
VALUES
	('a'),
	('En attente'),
	('e');
SELECT * FROM commande;
```

	commandeId	statut
1	2	Acceptée
2	3	En attente
3	4	Expédiée

Est-ce bon d’utiliser des triggers ?

- Tout code lancé, ici un trigger, bloque des enregistrements ce qui peut poser des problèmes d’interblocage ou de performances
- Vous risquez d’être tenté d’utiliser des curseurs. Attention, ils sont lents
- « Quand on a un marteau, tout devient un clou » : posez-vous la question de savoir si c’est le bon outil