POUR VOUS PROTÉGER ET PROTÉGER LES AUTRES,
RESPECTEZ LES MESURES D’HYGIÈNE ET GARDEZ VOS DISTANCES

Si vous êtes malade, restez chez vous
Lavez-vous les mains régulièrement
Gardez une distance de minimum 1,50 mètre avec les autres
Désinfectez vos surfaces de travail après usage (et avant)
Si vous êtes diagnostiqué positif à la covid, signalez-le à covid@ulb.be

Portez obligatoirement le masque
Respectez les sens de circulation, tenez votre droite et suivez les indications d’entrée et de sortie
Au cours d’une semaine, installez-vous dans la même zone de couleur de l’auditoire
Nettoyez régulièrement vos équipements

ulb.be/covid
INFO-H-415 - Advanced Databases

Session 1
Active Databases

Université libre de Bruxelles
École polytechnique de Bruxelles
Practicalities

• 12 exercise sessions:
  • presental? room indicated in Gehol
    http://gehol.ulb.ac.be
  • at distance?

• Presental sessions start at 14h15 and end at 15h45
• Don’t forget to clean your desk before and after the session

• 1 project (starting at week 8)
12 exercise sessions

- Sessions 1 – 3: *Active* databases
- Sessions 4 – 6: *Temporal* databases
- Sessions 7 – 9: *Graph* databases (?)
- Sessions 10–12: *Spatial* databases
Practicalities

Course’s Wiki

http://cs.ulb.ac.be/public/teaching/infoh415

Teaching Assistant

Dejaegere Gilles
Gilles.Dejaegere@ulb.ac.be
Evaluation

- 25% for the project,
- 75% for the written examination
Do you have questions?
Active Databases

SQL Server Triggers
A database trigger is **procedural code** that is automatically executed in response to certain **events** on a particular table or view in a database.

The trigger is mostly used for maintaining the **integrity** of the information on the database.
In SQL Server, triggers are executed directly after an **instruction** (i.e. not after each row or each transaction).

<table>
<thead>
<tr>
<th>SSN</th>
<th>Lab</th>
<th>Salary</th>
</tr>
</thead>
<tbody>
<tr>
<td>6789</td>
<td>1</td>
<td>30 000</td>
</tr>
<tr>
<td>5555</td>
<td>2</td>
<td>40 000</td>
</tr>
<tr>
<td>4321</td>
<td>1</td>
<td>43 000</td>
</tr>
<tr>
<td>7777</td>
<td>4</td>
<td>25 000</td>
</tr>
</tbody>
</table>

**SQL Example:**

```sql
UPDATE Employee
SET Salary = 0
WHERE Lab = 1;
```
SQL Server trigger types

- **AFTER triggers** are executed after the instruction takes place.

- **INSTEAD OF triggers** do not execute the triggering instruction, but executes custom code in place of it.
SQL Server triggers

Syntax

create trigger <name>
on <table>
{after|instead of} <list of events>
as
<transact-SQL-statements>

Possible events: insert, delete, update
SQL Server triggers

Inside the `<transact-SQL-statements>`, special tables allow accessing the newly created and the deleted rows.

Special tables

- **Inserted**: new or updated rows of the triggering transaction
- **Deleted**: deleted rows (or old state for updates) of the triggering transaction

Note that, since the trigger is executed at instruction level, these tables can contain many rows.
SQL Server triggers

### Employee

<table>
<thead>
<tr>
<th>SSN</th>
<th>Lab</th>
<th>Salary</th>
</tr>
</thead>
<tbody>
<tr>
<td>6789</td>
<td>1</td>
<td>30 000</td>
</tr>
<tr>
<td>5555</td>
<td>2</td>
<td>40 000</td>
</tr>
<tr>
<td>4321</td>
<td>1</td>
<td>43 000</td>
</tr>
<tr>
<td>7777</td>
<td>4</td>
<td>25 000</td>
</tr>
</tbody>
</table>

### Inserted

<table>
<thead>
<tr>
<th>SSN</th>
<th>Lab</th>
<th>Salary</th>
</tr>
</thead>
<tbody>
<tr>
<td>6789</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>4321</td>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>

### Deleted

<table>
<thead>
<tr>
<th>SSN</th>
<th>Lab</th>
<th>Salary</th>
</tr>
</thead>
<tbody>
<tr>
<td>6789</td>
<td>1</td>
<td>30 000</td>
</tr>
<tr>
<td>4321</td>
<td>1</td>
<td>43 000</td>
</tr>
</tbody>
</table>

```sql
UPDATE Employee
SET Salary = 0
WHERE Lab = 1;
```
Two possible actions

When a constraint violation is detected, two types of actions are possible:

**Abort**
The transaction is cancelled with a *rollback* statement and an error is raised.

**Repair**
An *update* statement modifies the database to make it consistent with the integrity constraints.
Example of a trigger

Consider two relations:

- **Employee** \((\text{Name}, \text{Salary}, \text{Department})\) with Department referencing **Department**.DeptNo
- **Department** \((\text{DeptNo}, \text{Manager})\) with Manager referencing **Employee**.Name

We want to ensure that the salary of an employee cannot be greater than that of his manager.

What are the events that could bring this rule to be violated?
Example of a trigger

• Employee (Name, Salary, Department)
• Department (DeptNo, Manager)

We want to ensure that the salary of an employee cannot be greater than that of his manager.

Constraint violating events:

• When adding an employee
• When modifying an employee’s salary
• When modifying an employee’s department
• When modifying department’s manager
Example of an aborting after insert trigger

```
cREATE TRIGGER Emp-insertion-abort
ON Employee
AFTER INSERT
AS

IF EXISTS(
    SELECT *
    FROM Inserted I,
        Department D,
        Employee Mgr
    WHERE I.DeptNo = D.DeptNo
    AND D.Manager = Mgr.Name
    AND Mgr.Salary < I.Salary )
BEGIN
    RAISERROR 13000 'The salary of an employee cannot be greater than that of his manager'
    ROLLBACK
END
```
Active Databases

Exercises
Training on your own machine:

- Download the SQL Server Management Studio here
- Download SQL Server Express here
Connecting to the database environment

• Do not hesitate to work in small groups (2-3)
  Be careful that every member has coding time!
  It is not enough to understand what a team mate does

• Boot the computer with Windows

• Log on to the computer with your netid

• Open SQL Server Management Studio

• Connect to the server “WIT-SQL-EDU”
  (using Windows authentication)
Loading the data set

Available on the labs web page:

http://cs.ulb.ac.be/public/teaching/infoh415/tp

Set-up

• Create a “infoh415-<your-netid>-PhD” database
  (drop it if it already exists)
• Open and run phd_createtable.sql
• Open and run phd_dbload.sql
  Caution: Select the right database before running these scripts!
  (see next slide)
Select the right database

Select the database **you created** either:

- using the client

- by starting your script by:

  ```
  use database_name
  ```
Practical steps for the exercises

We suppose that the database is initially consistent.

Steps

1. Determine when a constraint can be violated.
2. Then, decide on an action to be taken: *abort* or *repair*.
3. Write the trigger.
4. Test the trigger, by editing the data in a way that violates the constraint.