INSA INSTITUT NATIONAL DES SCIENCES APPLIQUÉES ROUEN NORMANDIE

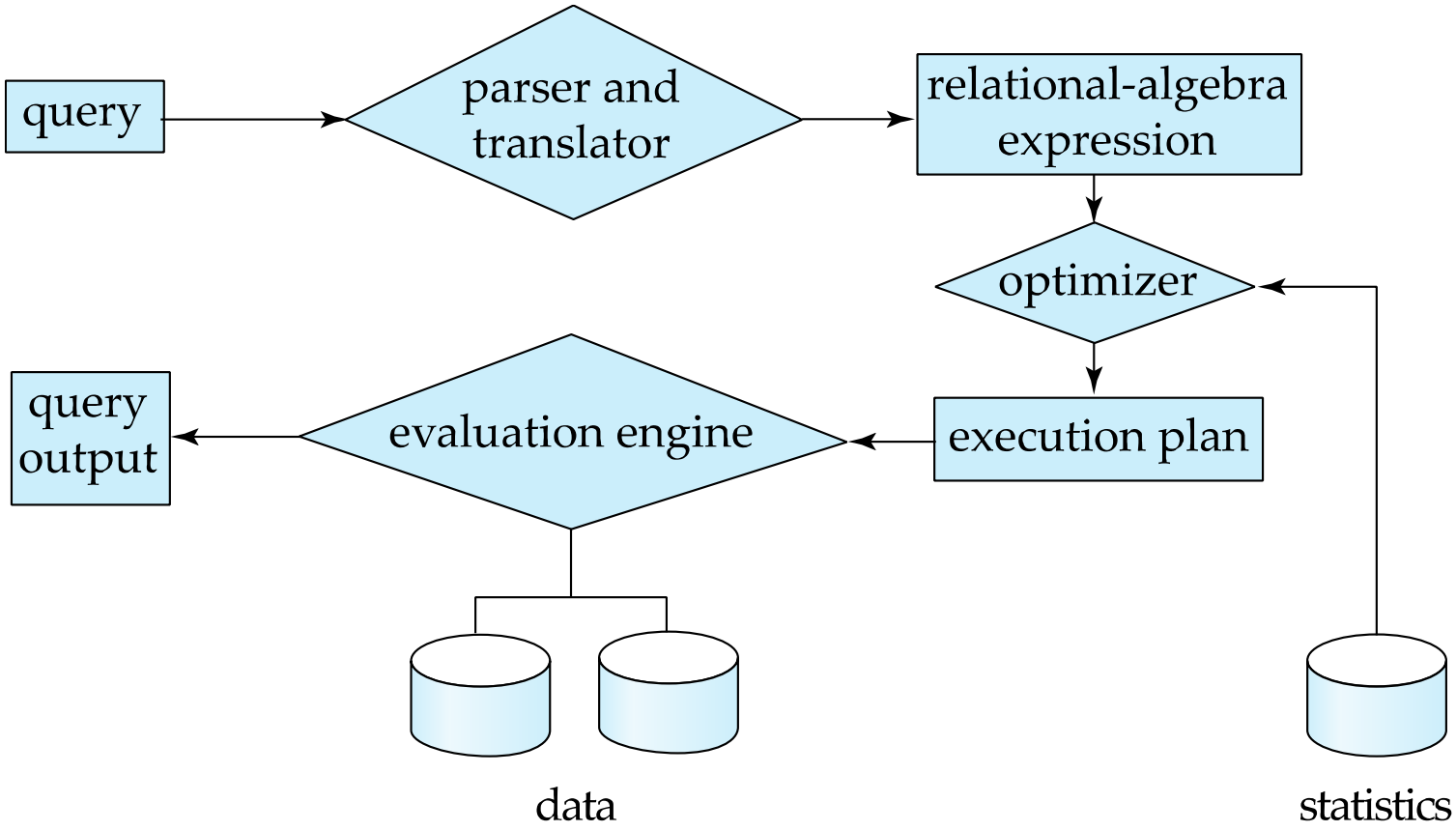
OPTIMISATION

PLAN D'EXECUTION

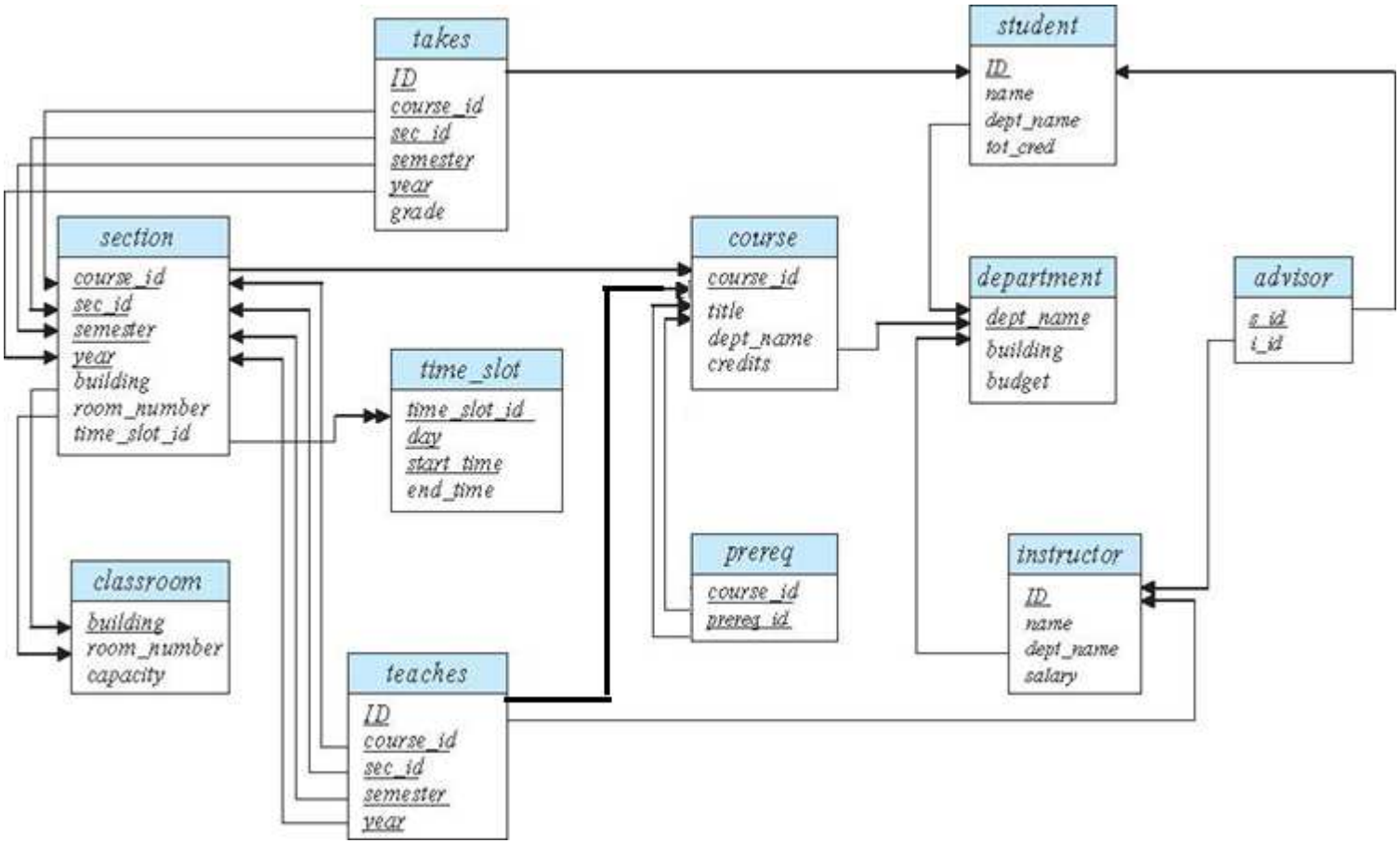
Mettre en œuvre les scripts SQL déclaratifs

Basic Steps in Query Processing

1. **Parsing and translation**
2. **Optimization**
3. **Evaluation**



Optimisation – exemple YALE



Optimisation – exemple YALE

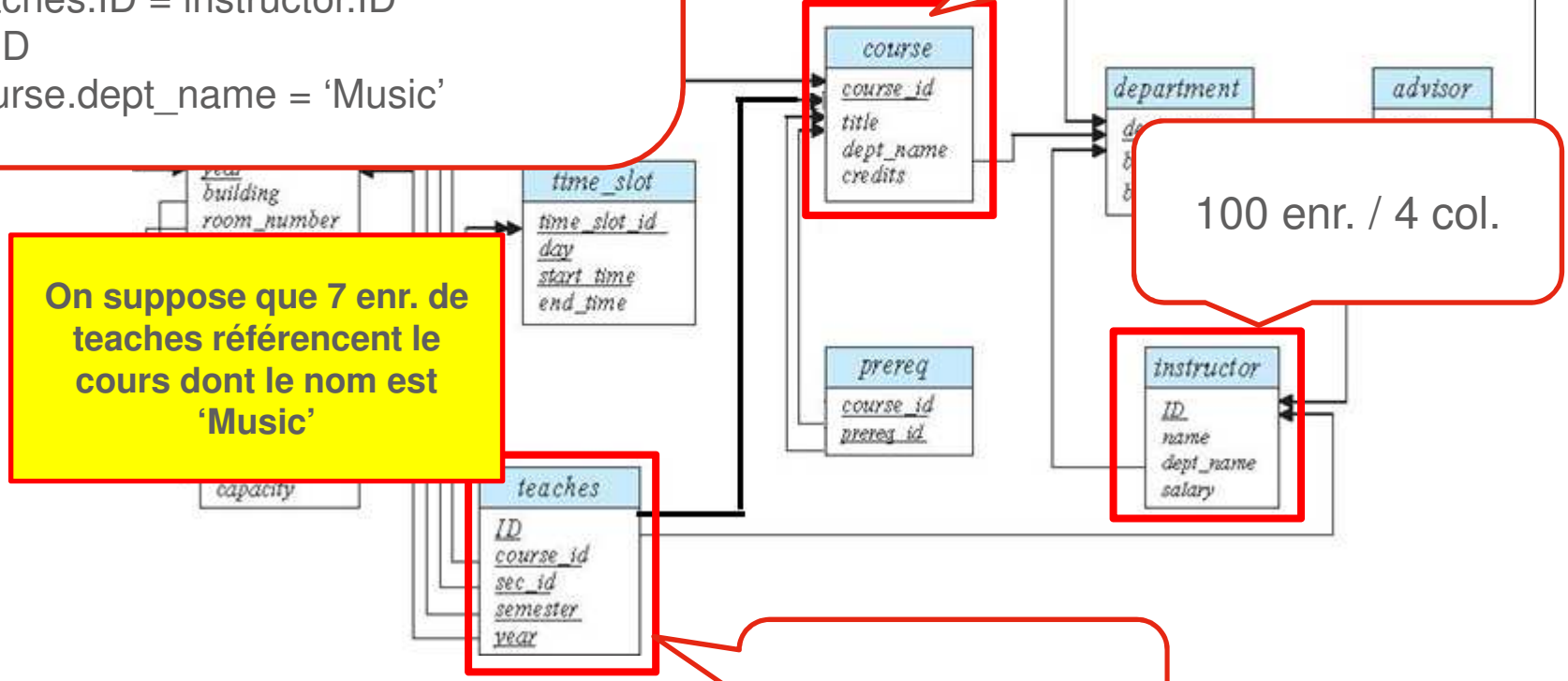

```
SELECT course.title, instructor.name
FROM course, instructor, teaches
WHERE
teaches.course_id  = course.course_ID
AND
teaches.ID = instructor.ID
AND
course.dept_name = 'Music'
```

4 col. / 2000 enr.

100 enr. / 4 col.

On suppose que 7 enr. de teaches référencent le cours dont le nom est 'Music'

5 col. / 2200 enr.

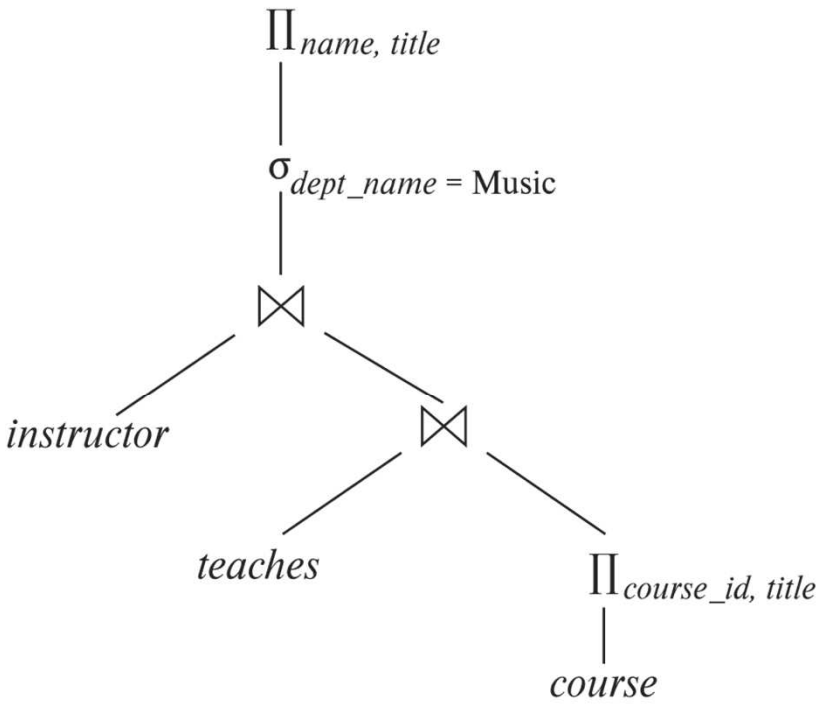


- **Algèbre relationnelle : langage procedural consistant en une suite d'opérations prenant 1 ou 2 relations en entrée et produisant une nouvelle relation en sortie.**
- **Six opérateurs de base**
  - **select: σ**
  - **project: ∏**
  - **union: ∪**
  - **set difference: –**
  - **Cartesian product: x**
  - **rename: ρ**
- **Quatre opérateur composé (à partir de la base)**
  - **Join: ⋈**
  - **Join left : ⟕**
  - **Join right : ⟖**
  - **Join full : ⟗**

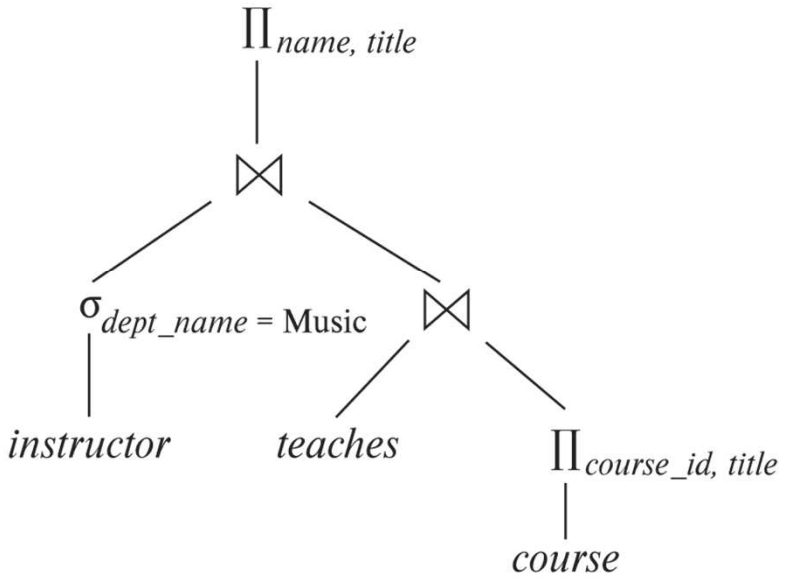
***join* (jointure) : X (produit cartésien) combiné à un σ *(sélection)* sur les colonnes communes.**

Optimisation – exemple (2)

- SELECT instructor.name, course.title FROM course, instructor, teaches
- WHERE teaches.course_id = course.course_ID
- AND teaches.ID = instructor.ID
- ANDcourse.dept_name = ‘Music’



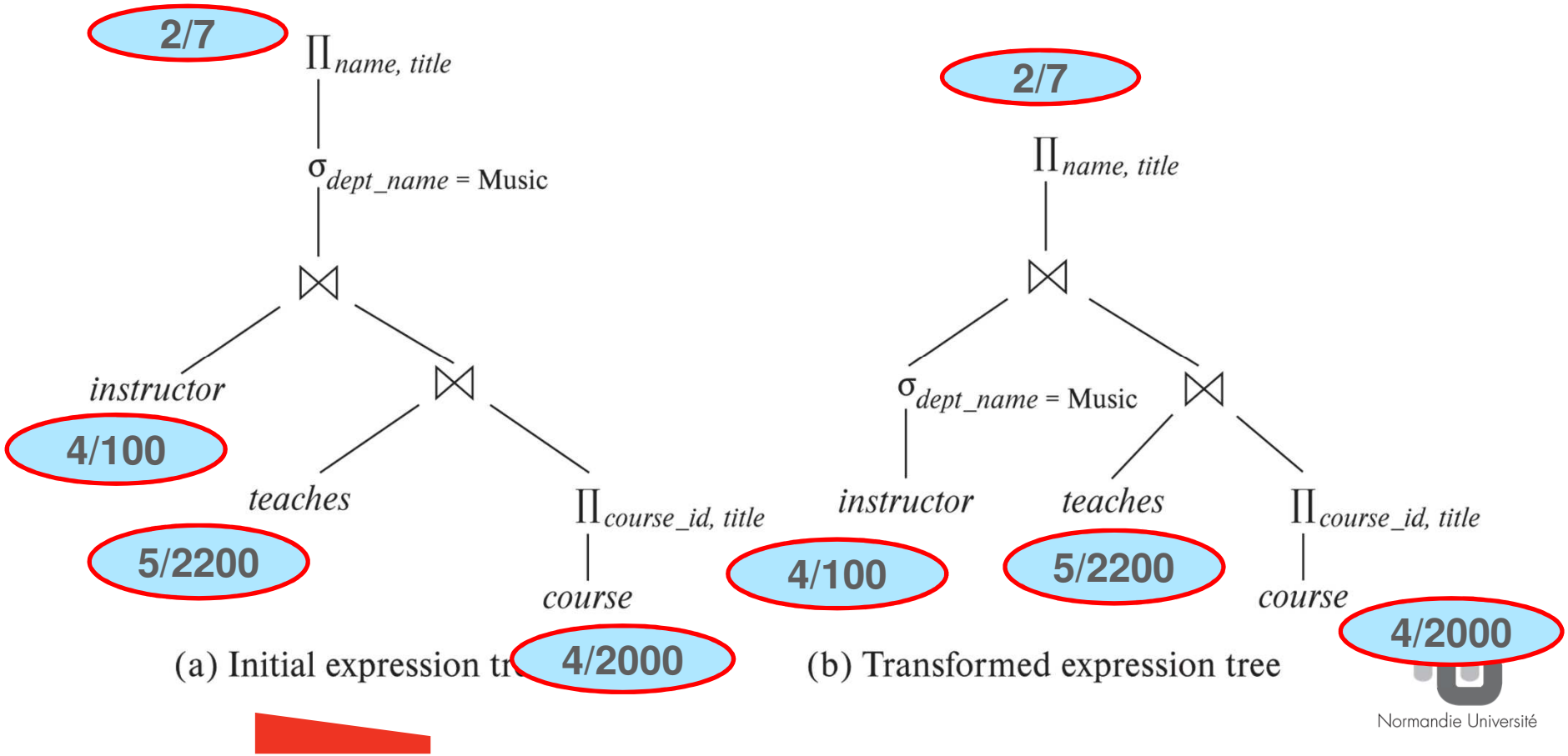
(a) Initial expression tree



(b) Transformed expression tree

Optimisation – exemple (2)

- SELECT instructor.name, course.title FROM course, instructor, teaches
- WHERE teaches.course_id = course.course_ID
- AND teaches.ID = instructor.ID
- ANDcourse.dept_name = 'Music'

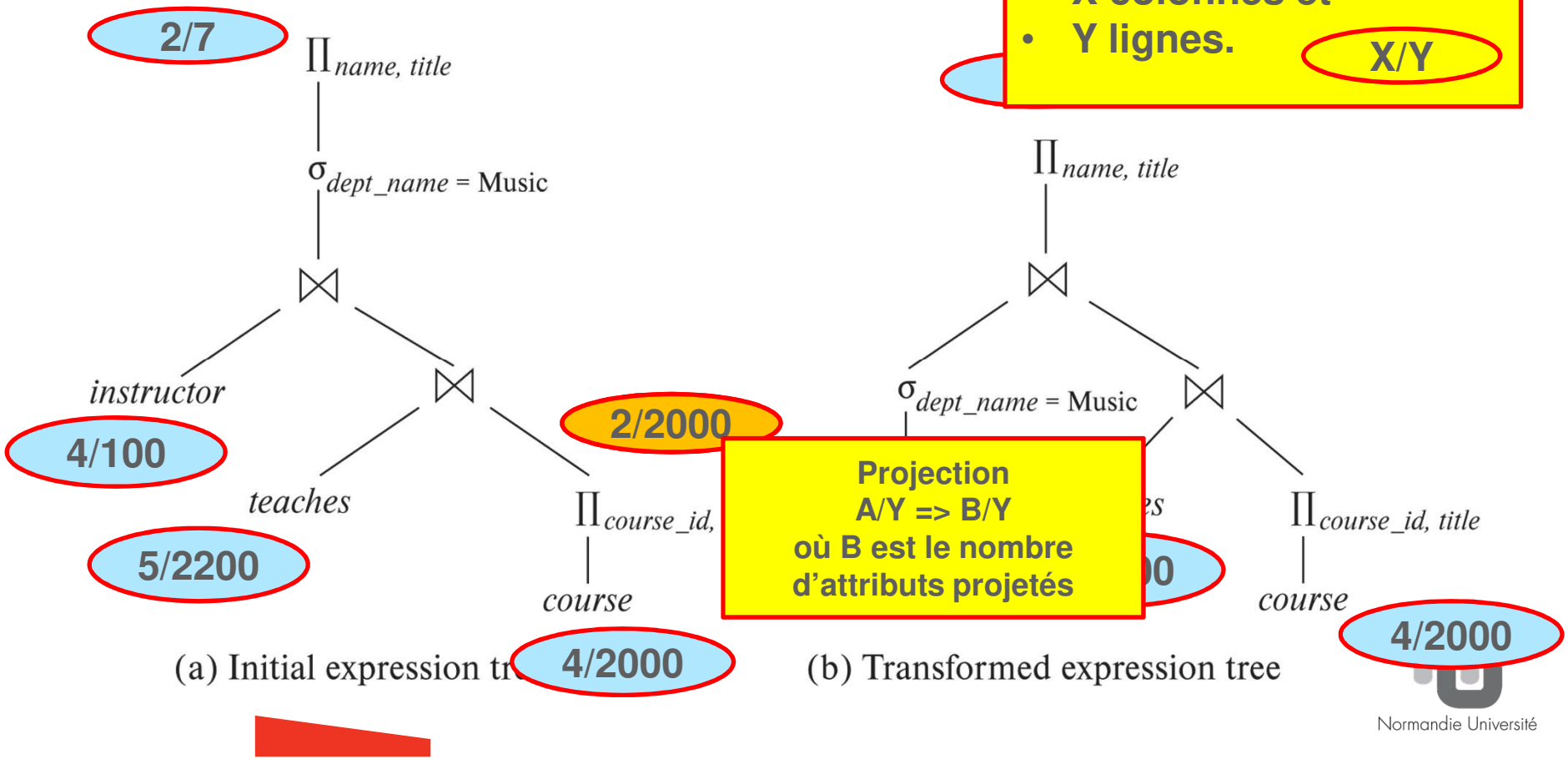


Optimisation – exemple (2)

- SELECT instructor.name, course.title FROM course, instructor, teaches
- WHERE teaches.course_id = course.course_ID
- AND teaches.ID = instructor.ID
- ANDcourse.dept_name = 'Music'

Table intermédiaire avec
- **X colonnes et**
- **Y lignes.**

X/Y



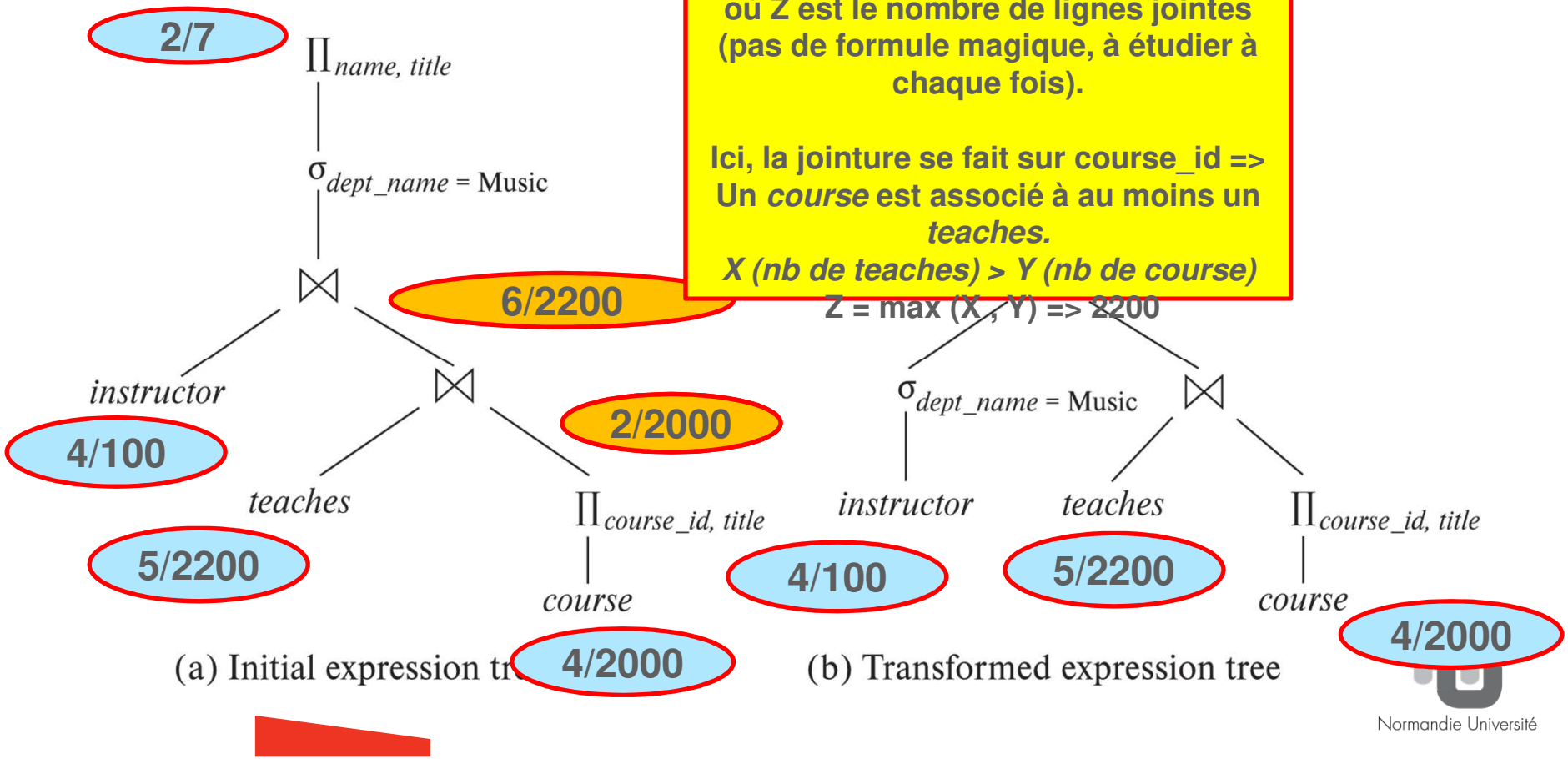
Projection
A/Y => B/Y
où B est le nombre d'attributs projetés

Optimisation – exemple (2)

- SELECT instructor.name, course.title FROM course, instructor, teaches
- WHERE teaches.course_id = course.course_ID
- AND teaches.ID = instructor.ID
- ANDcourse.dept_name = 'Mu

Jointure
A/X et B/Y => A+B-1/Z
où Z est le nombre de lignes jointes (pas de formule magique, à étudier à chaque fois).

Ici, la jointure se fait sur course_id => Un *course* est associé à au moins un *teaches.*
X (nb de teaches) > Y (nb de course)
Z = max (X , Y) => 2200

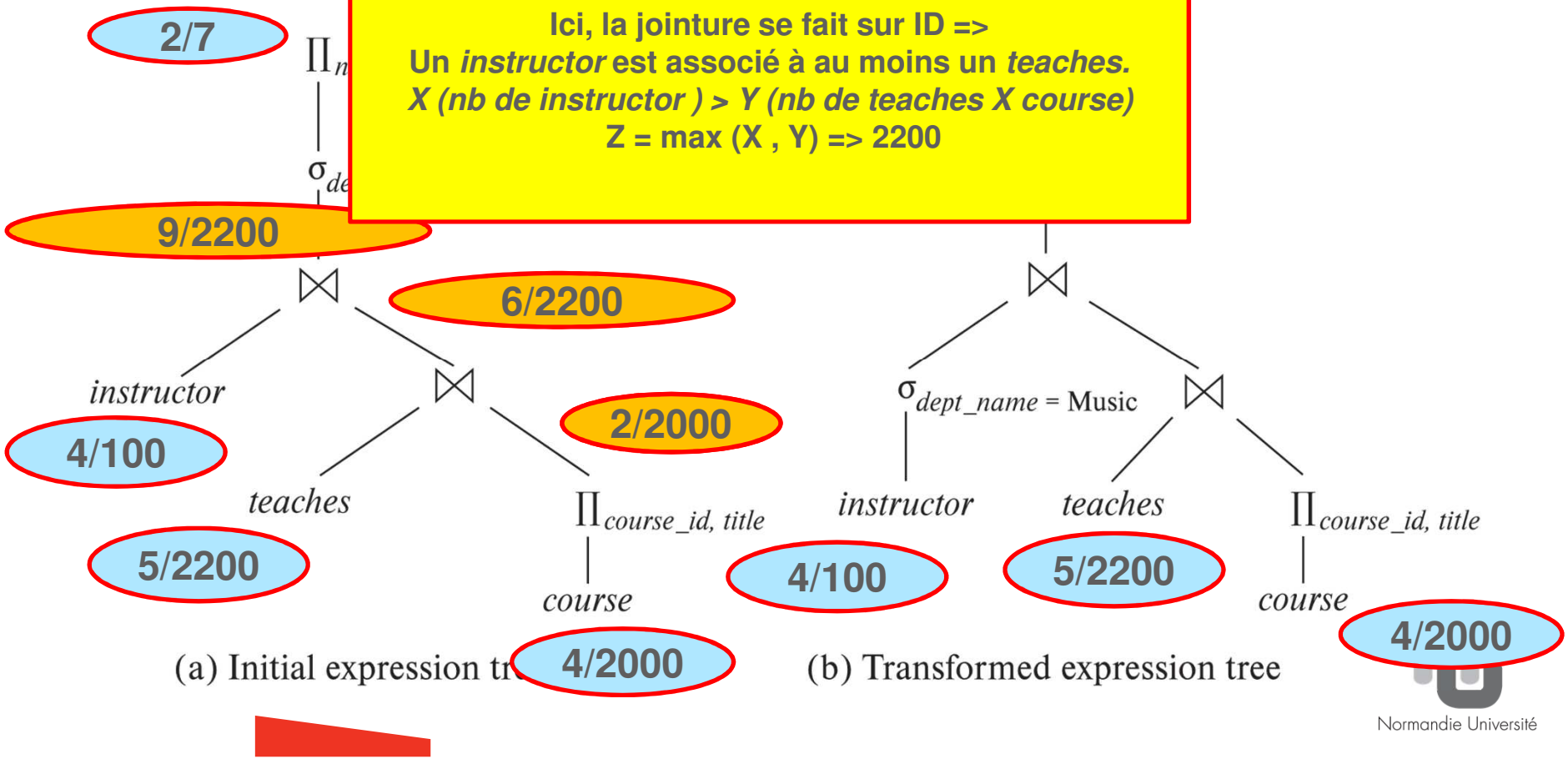


Optimisation – exemple (2)

- SELECT instructor.name, course.title FROM course, instructor, teaches
- WHERE teaches.course_id = course.course_ID
- AND teache
- ANDcourse.

Calcul du nb de colonnes : 9 (4 + 6 – 1)

Ici, la jointure se fait sur ID =>
Un *instructor* est associé à au moins un *teaches.*
X (nb de instructor) > Y (nb de teaches X course)
Z = max (X , Y) => 2200

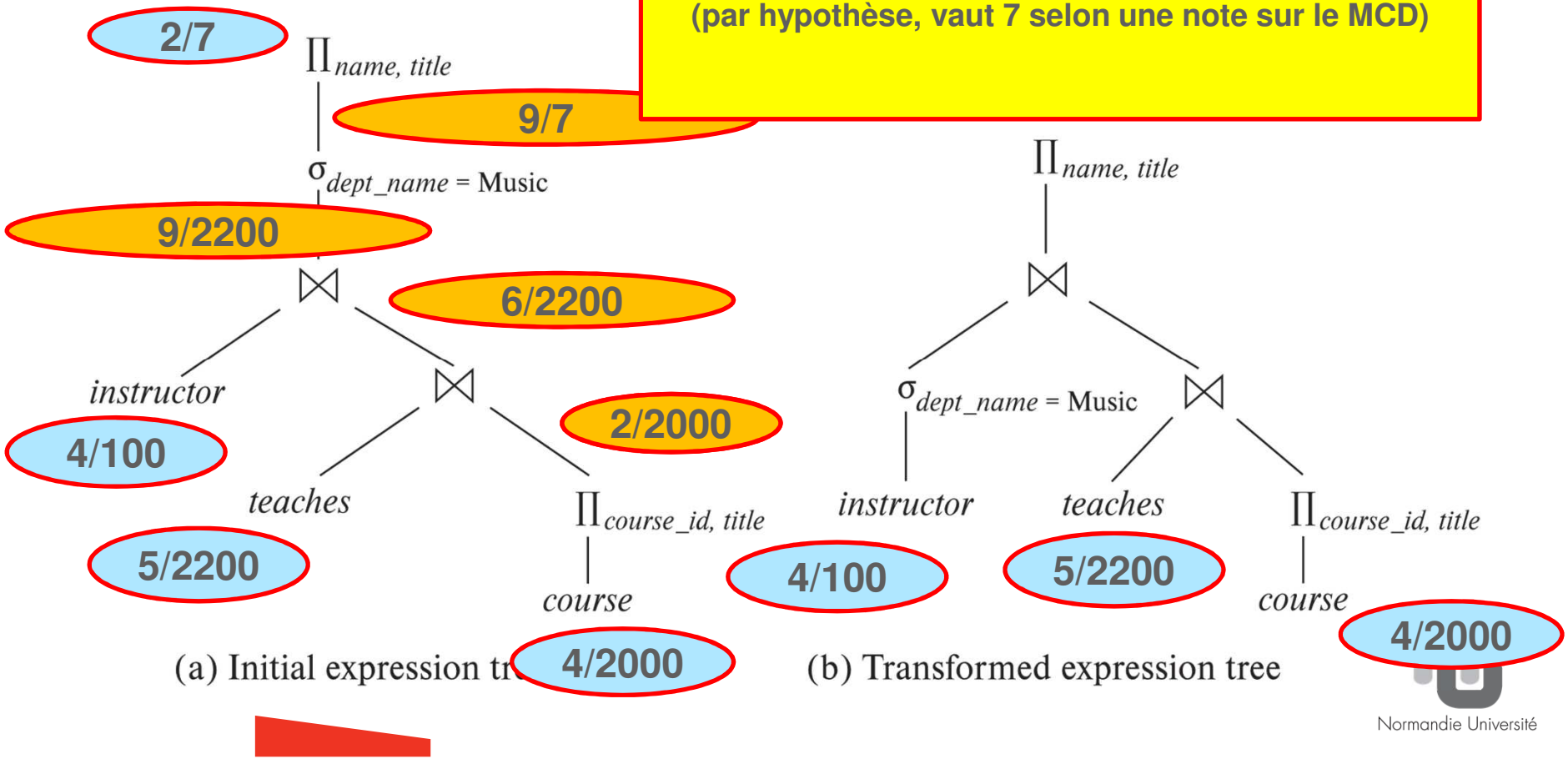


Optimisation – exemple (2)

- SELECT instructor.name, c
- WHERE teaches.course_id
- AND teaches.ID = instructo
- ANDcourse.dept_name = 'M

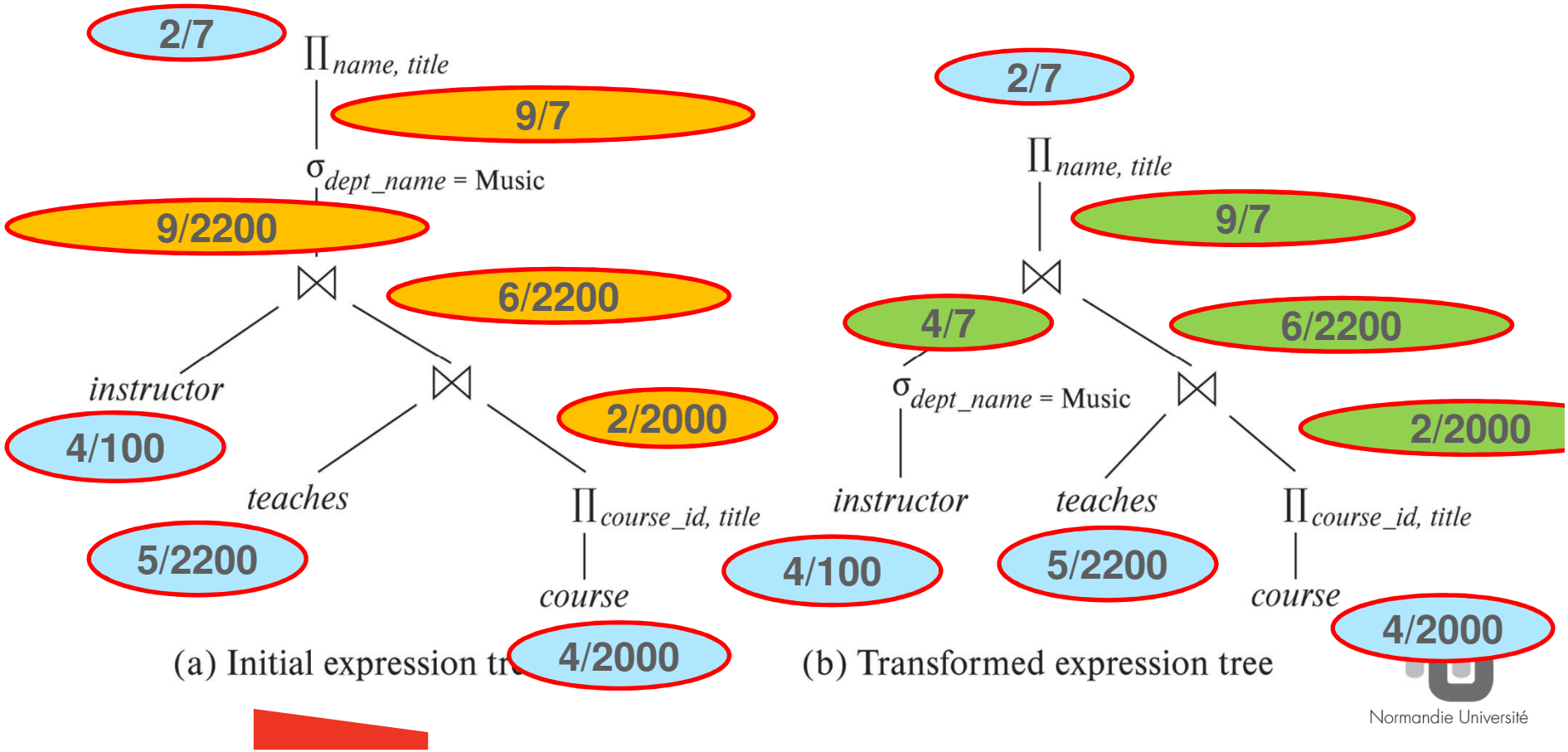
selection

On ne conserve que les teaches x course x instructor dont le dept_name vaut 'Music'.
(par hypothèse, vaut 7 selon une note sur le MCD)



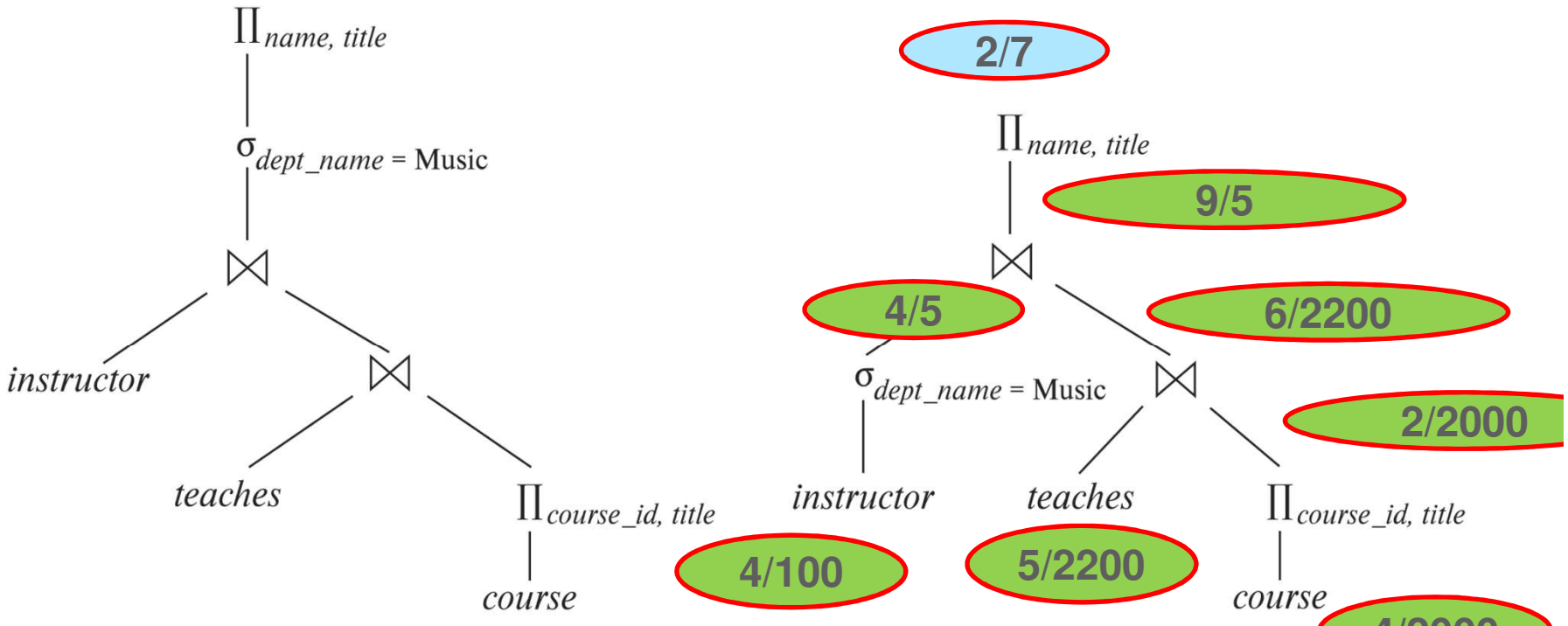
Optimisation – exemple (2)

- SELECT instructor.name, course.title FROM course, instructor, teaches
- WHERE teaches.course_id = course.course_ID
- AND teaches.ID = instructor.ID
- AND course.dept_name = 'Music'



Optimisation – exemple (2)

- SELECT instructor.name, course.title FROM course, instructor, teaches
- WHERE teaches.course_id = course.course_ID
- AND teaches.ID = instructor.ID
- ANDcourse.dept_name = 'Music'



(a) Initial expression tree

(b) Transformed expression tree

- **Pour observer le plan d'execution d'une requête sous Postgres**
  - **> EXPLAIN ANALYZE <requete> ;**

QUERY PLAN
Gather (cost=1000.00..7483.92 rows=1 width=4) (actual time=0.329..87.243 rows=1 loops=1)
Workers Planned: 1
Workers Launched: 1
-> Parallel Seq Scan on demo (cost=0.00..6483.82 rows=1 width=4) (actual time=37.892..79.399 rows=0 loops=2)
Filter: (id = 20)
Rows Removed by Filter: 500000
Planning Time: 0.158 ms
Execution Time: 87.272 ms