Lecture 12

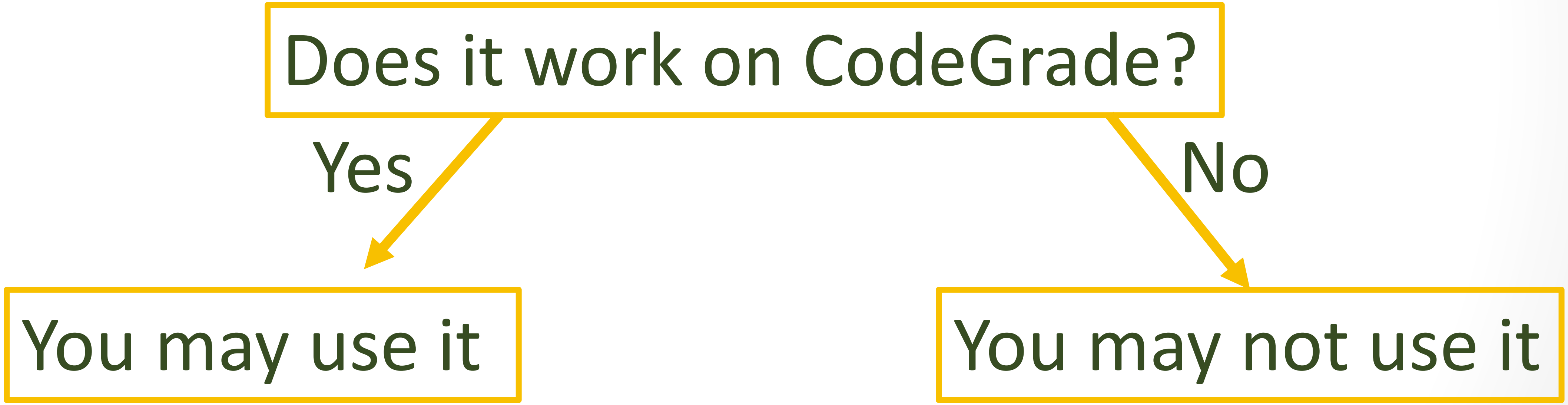
COMP207

Overview

- Relational algebra
- Query plans
- Do things faster
- Index

Practical SQL assignment FAQ part 2

- The most common question is still: Am I allowed to use this method not covered in the course?
  - Use the following flowchart to figure it out:



Does it work on CodeGrade?

Yes

No

You may use it

You may not use it

Algebra

- An algebra is about abstract sets that allow operations such as multiplication, addition and scalar multiplication
  - The key thing is that given one or two objects (and sometimes other things), you can produce a new object
  - E.g. if you have the object 3, the object 4, then the + operation between them can produce the object 3+4=7
- It does not need to be numbers

Relational algebra

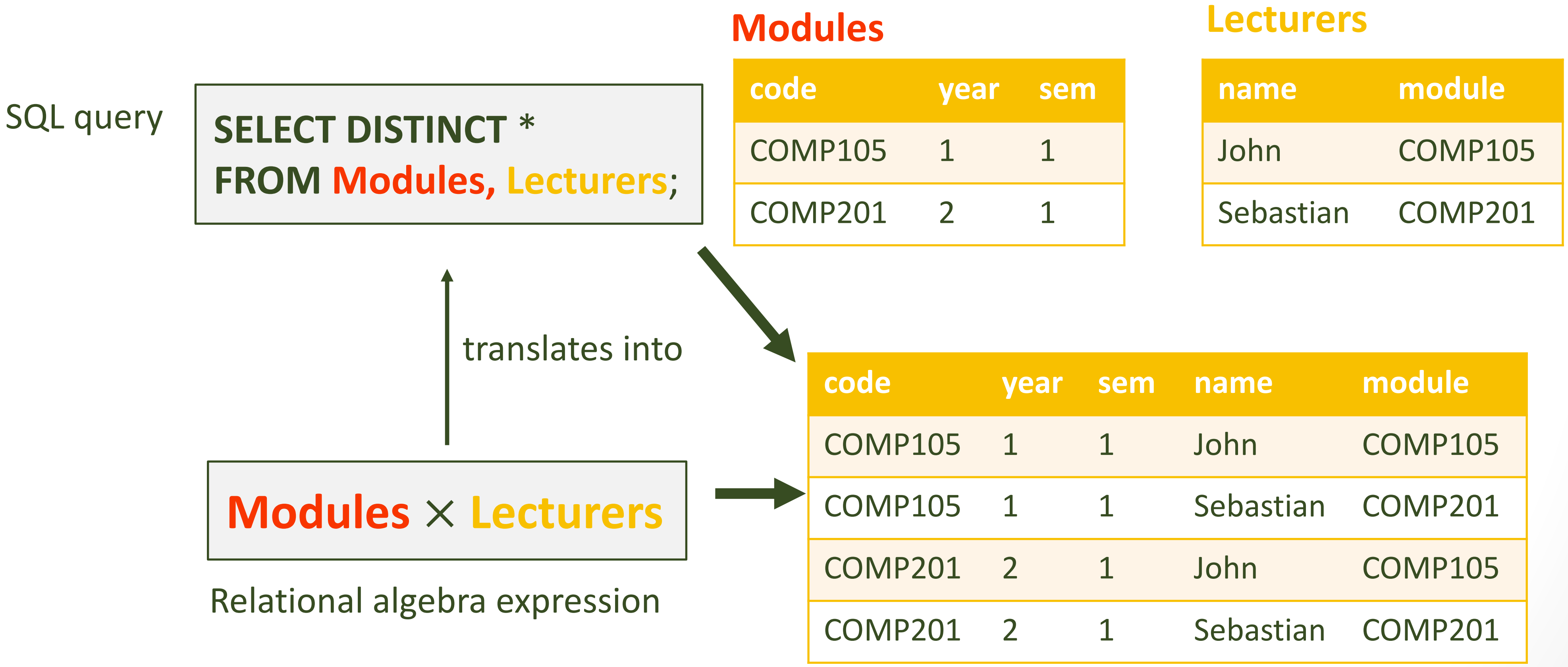
- Algebra, but with tables instead of other objects

- 3+4 = 7

- Table R cross-product table S = a table, that for each pair of rows r,s, with r in R and s in S, has a row (r,s)

Cartesian Product (×)

- $R_1 \times R_2$ = pairs each tuple in R_1 with each tuple in R_2



SQL query

```
SELECT DISTINCT *
FROM Modules, Lecturers;
```

translates into

Modules × Lecturers

Relational algebra expression

Modules

code	year	sem
COMP105	1	1
COMP201	2	1

Lecturers

name	module
John	COMP105
Sebastian	COMP201

code	year	sem	name	module
COMP105	1	1	John	COMP105
COMP105	1	1	Sebastian	COMP201
COMP201	2	1	John	COMP105
COMP201	2	1	Sebastian	COMP201

Natural Join (⋈)

Employees

birthday	first_name	family_name	e_id
1990-11-10	Anne	Smith	1
2000-02-05	David	Jones	2
1995-05-09	William	Taylor	3

- $R_1 \bowtie R_2$ = pairs each tuple in R_1 with each tuple in R_2 with matching common attributes

Transactions

t_id	c_id	e_id
1	3	1
2	6	1
3	19	3

```
SELECT DISTINCT *
FROM Employees NATURAL JOIN
Transactions;
```

translates into

Employees ⋈ Transactions

Relational algebra expression

birthday	first_name	family_ name	e_id	t_id	c_id
1990-11-10	Anne	Smith	1	1	3
1990-11-10	Anne	Smith	1	2	6
1995-05-09	William	Taylor	3	3	19

Projection (π)

- $\pi_{\text{attribute list}}(\mathbf{R})$ = restricts **R** to the attributes in **attribute list**

SQL query

```
SELECT DISTINCT   family_name,
                  birthday
FROM Employees;
```

translates into

$\pi_{\text{family\_name,birthday}}(\textbf{Employees})$

Relational algebra expression

Employees

birthday	first_name	family_name
1990-11-10	Anne	Smith
2000-02-05	David	Jones
1995-05-09	William	Taylor

family_name	birthday
Smith	1990-11-10
Jones	2000-02-05
Taylor	1995-05-09

Renaming (ρ)

- $\rho_{A1 \rightarrow B1, A2 \rightarrow B2, \ldots}(R)$ = renames attribute A1 to B1, attribute A2 to B2, …

SQL query

```
SELECT DISTINCT birthday AS bday, first_name, family_name
FROM Employees;
```

Employees

birthday	first_name	family_name
1990-11-10	Anne	Smith
2000-02-05	David	Jones
1995-05-09	William	Taylor

translates into

$\rho_{birthday \rightarrow bday}(Employees)$

Relational algebra expression

bday	first_name	family_name
1990-11-10	Anne	Smith
2000-02-05	David	Jones
1995-05-09	William	Taylor

Selection (σ)

The SELECT keyword in SQL has nothing to do with the selection operator!

$\sigma_{\text{condition}}(R)$ = set of all tuples in **R** that satisfy the **condition**

SQL query

```
SELECT DISTINCT *
FROM Employees
WHERE first_name='Anne';
```

translates into

$\sigma_{\text{first\_name='Anne'}}(\text{Employees})$

Relational algebra expression

Employees

birthday	first_name	family_name
1990-11-10	Anne	Smith
2000-02-05	David	Jones
1995-05-09	William	Taylor

birthday	first_name	family_name
1990-11-10	Anne	Smith

Index in MySQL

- CREATE INDEX indR USING BTREE ON R (value);
- CREATE INDEX indRHash USING HASH ON R (value);