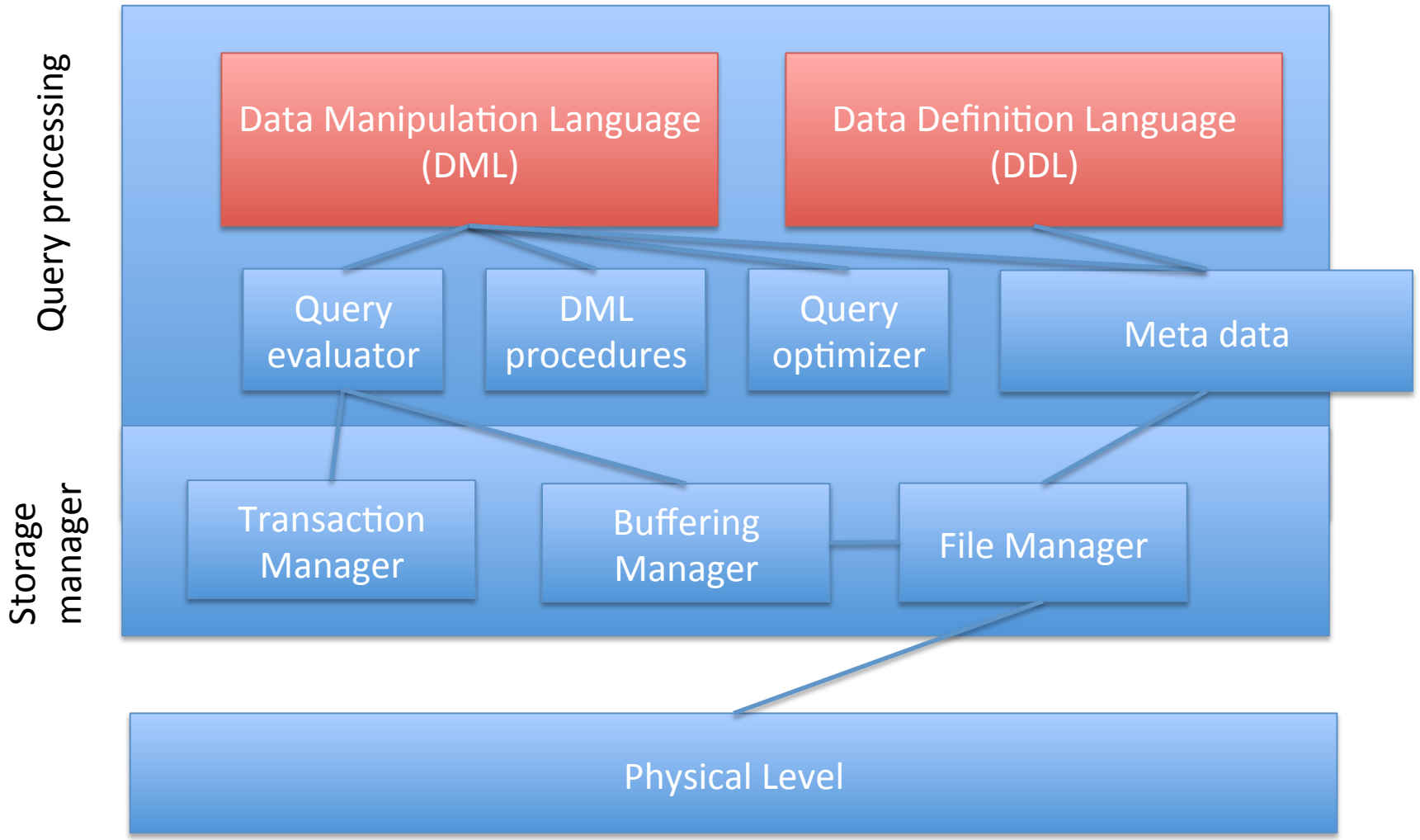
SQL

Carsten Schürmann

Database System Architecture



SQL

The most widely used relational query language.

- Major standard is SQL-1999 (=SQL3)
- Introduced “Object-Relational” concepts
- SQL 2003, SQL 2008 have small extensions
- SQL 92 is a basic subset

Data Manipulation Language DML

- Select – From – Where
- Renaming
- Set operation
- Ordering
- Aggregate functions
- Nested subqueries
- Insertion
- Joins

SQL Syntax

select A1, A2, ... An

from R1, R2, ... Rm

where P

Our Database for Today

Student

cpr	name	address
140298-1234	Jesper	Copenhagen
041297-5367	Nikoline	Aarhus
151197-2352	Claus	Dragør
050596-1142	Martin	Copenhagen

Takes

cpr	course	grade
140298-1234	SIDD	10
041297-5367	SIDD	12

Class

course	name	etcs
SIDD	Databases	7.5
CSYS	Critical Systems	7.5

Query

Find the cprs of all students named Claus!

$$\pi_{\text{cpr}}(\sigma_{\text{name = 'Claus'}}(\text{Student}))$$

```
select cpr
from   Student
where  name = "Claus"
```

General Form

$\pi_{A1,\ldots,An}(\sigma_{P}(R_1 \times R_2 \ldots R_m))$

```
select A1,…,An
from   R1,R2 …. Rm
where  P
```

Looking at the Condition

Find the cprs of all students named Claus from Copenhagen!

$$\pi_{\text{cpr}}(\sigma_{\text{name = 'Claus'} \wedge \text{ address = 'Copenhagen'}} \text{(Student)})$$

```
select  cpr
from    Student
where   name = "Claus"
and     address = "Copenhagen"
```

More logical operations

Boolean operators (`and or not` …)

Comparision operators (`< > =` …)

String operators (`like % _` …)

Date operators (`DATE, YEAR`, …)

Find all people at addresses starting with “D”

```
select cpr
from   Student
where  address like “D%”
```

Our Database for Today

Student

cpr	name	address
140298-1234	Jesper	Copenhagen
041297-5367	Nikoline	Aarhus
151197-2352	Claus	Dragør
050596-1142	Martin	Copenhagen

Takes

cpr	course	grade
140298-1234	SIDD	10
041297-5367	SIDD	12

Class

course	name	etcs
SIDD	Databases	7.5
CSYS	Critical Systems	7.5

Query

Find the names of all students taking SIDD!

```
select name
from   Student, Takes
where  Student.cpr = Takes.cpr
and    Takes.class = "SIDD"
```

Data Manipulation Language DML

- Select – From – Where
- Renaming
- Set operation
- Ordering
- Aggregate
- Nested subqueries
- Insertion
- Joins

Explicit Renaming as Shorthand

Find the names of all students taking SIDD!

```
select name
from   Student as S, Takes as T
where  S.cpr = T.cpr
and    T.class = "SIDD"
```

Explicit Renaming for Self-Join

Who are Jesper's grand-parents?

Parent

cpr	parent	child
140298-1234	Jesper	Lotte
041297-5367	Nikoline	Jesper
151197-2352	Claus	Nikoline
050596-1142	Martin	Lotte

```
select S.name
from   Parent as S, Parent as T
where  S.child = T.parent
and    T.child = "Jesper"
```

Explicit Renaming for Self-Join

Find courses with more ECTS than SIDD

```
select S.course
from   Class as S, Class as T
where  S.ects > T.ects
and    T.course = "SIDD"
```

Data Manipulation Language DML

- Select – From – Where
- Renaming
- Set operation
- Ordering
- Aggregate
- Nested subqueries
- Insertion
- Joins

Our Database for Today

Student

cpr	name	address
140298-1234	Jesper	Copenhagen
041297-5367	Nikoline	Aarhus
151197-2352	Claus	Dragør
050596-1142	Martin	Copenhagen

Takes

cpr	course	grade
140298-1234	SIDD	10
041297-5367	SIDD	12

Class

course	name	etcs
SIDD	Databases	7.5
CSYS	Critical Systems	7.5

Set operations

Find the CPRs of people taking both SIDD and Critical Systems

```
(select cpr
 from   takes
 where  course = "SIDD")
intersect
(select cpr
 from   takes
 where  course = "CSYS")
```

Set operations

Find the CPRs of people taking SIDD or Critical Systems

```
(select cpr
 from   takes
 where  course = "SIDD")
union
(select cpr
 from   takes
 where  course = "CSYS")
```

Set operations

Find the CPRs of people taking both SIDD but not Critical Systems

```
(select cpr
 from   takes
 where  course = "SIDD")
except
(select cpr
 from   takes
 where  course = "CSYS")
```

Data Manipulation Language DML

- Select – From – Where
- Renaming
- Set operation
- Ordering
- Aggregate
- Nested subqueries
- Insertion
- Joins

Ordering

Find all students sorted by name!

```
select *
from Student
order by name asc
```

`asc` is default

Ordering

Find all students lexicographically sorted by name and cpr (in reverse)

```
select *
from Student
order by name, cpr desc
```

Data Manipulation Language DML

- Select – From – Where
- Renaming
- Set operation
- Ordering
- Aggregates
- Nested subqueries
- Insertion
- Joins

Aggregates

- How to compute the average grade for SIDD?
- How to compute how many students are taking SIDD?
- How to compute the worst grade CSYS?

We are leaving the confines of relational algebra!

Aggregates

How to compute the average grade for SIDD?

Takes

cpr	course	grade
140298-1234	SIDD	10
041297-5367	SIDD	12
140789-7612	SIDD	10
140789-7612	CSYS	4
041297-5367	CSYS	10
010292-3333	SIDD	4

```
select avg(grade)
from Takes
Where course = "SIDD"
```

avg(grade)
9

Aggregrates

- Average `avg(grade)`
- Counting `count(grade)`
- Minimum `min(grade)`
- Maximum `max(grade)`

None of these aggregrate functions is expressible in relational algebra.

Aggregates

Find the total number of students for each course

Takes

cpr	course	grade
140298-1234	SIDD	10
041297-5367	SIDD	12
140789-7612	SIDD	10
140789-7612	CSYS	4
041297-5367	CSYS	10
010292-3333	SIDD	4

```
select course, count(*)
from Takes
group by course
```

course	count(*)
SIDD	4
CSYS	2

Aggregates - Renaming

Find the total number of students for each course

Takes

cpr	course	grade
140298-1234	SIDD	10
041297-5367	SIDD	12
140789-7612	SIDD	10
140789-7612	CSYS	4
041297-5367	CSYS	10
010292-3333	SIDD	4

```
select course, count(*) as size
from Takes
group by course
```

course	size
SIDD	4
CSYS	2

Aggregates

What is the grade average of each student?

Takes

cpr	course	grade
140298-1234	SIDD	10
041297-5367	SIDD	12
140789-7612	SIDD	10
140789-7612	CSYS	4
041297-5367	CSYS	10
010292-3333	SIDD	4

```
select cpr, avg(grade)
from Takes
group by cpr
```

cpr	avg(grade)
140298-1234	10
041297-5367	11
140789-7612	7
010292-3333	4

Aggregates

Which students have an average grade above 10?

Takes

cpr	course	grade
140298-1234	SIDD	10
041297-5367	SIDD	12
140789-7612	SIDD	10
140789-7612	CSYS	4
041297-5367	CSYS	10
010292-3333	SIDD	4

```
select cpr, avg(grade)
from Takes
group by cpr
having avg(grade)>10
```

cpr	avg(grade)
140298-1234	10
041297-5367	11
140789-7612	7
010292-3333	4

Refined SQL Syntax

select A1, A2, ... An

from R1, R2, ... Rm

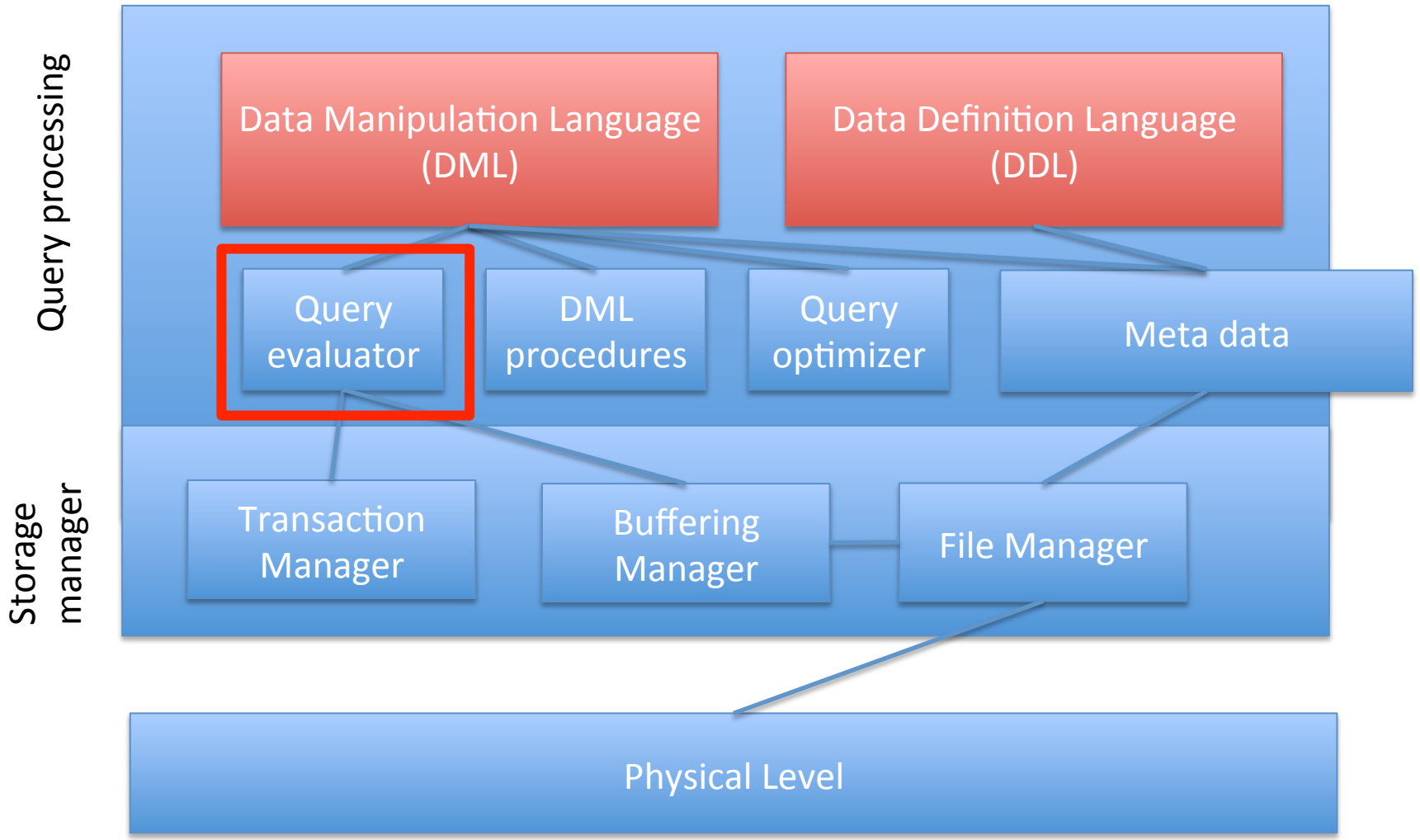
where P

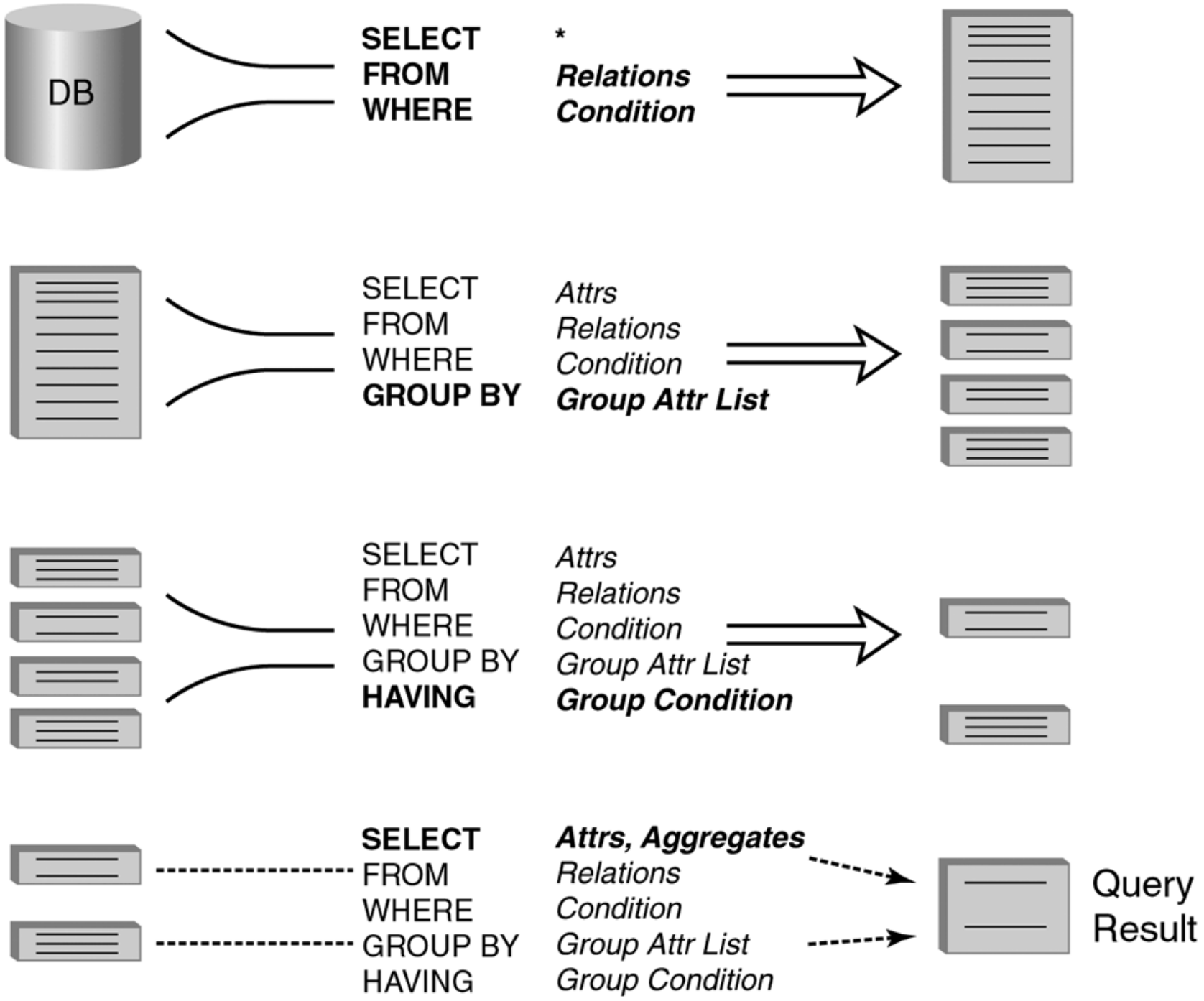
order by B1, B2, ... Bn

group by C1, C2, ... Cn

having Q

Database System Architecture




DB
SELECT
FROM
WHERE
*
Relations
Condition
SELECT
FROM
WHERE
GROUP BY
Attrs
Relations
Condition
Group Attr List
SELECT
FROM
WHERE
GROUP BY
HAVING
Attrs
Relations
Condition
Group Attr List
Group Condition
SELECT
FROM
WHERE
GROUP BY
HAVING
Attrs, Aggregates
Relations
Condition
Group Attr List
Group Condition
Query
Result

Data Manipulation Language DML

- Select – From – Where
- Renaming
- Set operation
- Ordering
- Aggregates
- Nested subqueries
- Insertion
- Joins

Our Database for Today

Student

cpr	name	address
140298-1234	Jesper	Copenhagen
041297-5367	Nikoline	Aarhus
151197-2352	Claus	Dragør
050596-1142	Martin	Copenhagen

Takes

cpr	course	grade
140298-1234	SIDD	10
041297-5367	SIDD	12

Class

course	name	etcs
SIDD	Databases	7.5
CSYS	Critical Systems	7.5

Nested Subqueries

Find all names of students who take SIDD.

Alternatively

Find the names for the CPRs that are contained in the set of of CPRs of students who take SIDD.

Nested Subqueries

Find the names for the CPRs that are contained in the set of of CPRs of students who take SIDD.

```
select name
from Student
where cpr "is contained in"
        select cpr
        from Takes
        where course = "SIDD"
```

Nested Subqueries

Find the names for the CPRs that are contained in the set of of CPRs of students who take SIDD.

```
select name
from Student
where cpr in
        (select cpr
         from Takes
         where course = "SIDD")
```

That's Not New …

Find the names for the CPRs that are contained in the set of of CPRs of students who take SIDD.

```
select name
from Student, Takes
where Student.cpr = Takes.cpr
and course = "SIDD"
```

…. But Convenient

Nested Subqueries

Find the student with the highest CPR number.

Option #1

Nested Subqueries

Find the student with the highest CPR number.

```
select name
from Student
where cpr "is greater or equal to any cpr in"
          ( select cpr
            from Takes
          )
```

Nested Subqueries

Find the student with the highest CPR number.

```
select name
from Student
where cpr >= all
          ( select cpr
            from Takes
          )
```

Nested Subqueries

Find the student with the highest CPR number without nested subqueries.

Option #2

$100 question

Find the student with the highest CPR number without nested queries!

Idea: Find all but the highest CPR!

```
select S.cpr, S.name, S.address
from Student as S, Student as T
where S.cpr < T.cpr
```

$100 question

Find the student with the highest CPR number without nested queries!

Idea: Take all students except those from before

```
(select * from Student)
except
(select S.cpr, S.name, S.address
 from Student as S, Student as T
 Where S.cpr < T.cpr)
```

Nested Subqueries

Find the student with the highest CPR number using aggregates.

Option #3

Nested Subqueries

Find the student with the highest CPR number.

```
select name
from Student
where cpr is equal to
        ( select max(cpr)
          from Takes
        )
```

Nested Subqueries

Find the student with the highest CPR number.

```
select name
from Student
where cpr in
        ( select max(cpr)
          from Takes
        )
```

Because of what we already discussed!

Nested Subqueries

Find the student with the highest CPR number.

```
select name
from Student
where cpr =
        ( select max(cpr)
          from Takes
        )
```

Because this relation contains only one element

Exercise

Find the CPR number of the student with the highest average grade.

```
select cpr, avg(grade)
from Takes
group by cpr
having avg(grade)  is the highest ever
```

Exercise

Find the CPR number of the student with the highest average grade

```
select cpr, avg(grade)
from Takes
group by cpr
having avg(grade) >= all
     (select avg(grade)
      from Takes
      group by cpr)
```

Relation of all grade averages

Nesting Operators

- `A < all (R)` A is less than every tuple in R
- `A <> all (R)` A not in R
- `A > some (R)` A is greater than a tuple in R
- `A = some (R)` A is in R
- `Exist (R)` checks if R is inhabited

Our Database for Today

Student

cpr	name	address
140298-1234	Jesper	Copenhagen
041297-5367	Nikoline	Aarhus
151197-2352	Claus	Dragør
050596-1142	Martin	Copenhagen

Takes

cpr	course	grade
140298-1234	SIDD	10
041297-5367	SIDD	12

Class

course	name	etcs
SIDD	Databases	7.5
CSYS	Critical Systems	7.5

Exists

Find all courses nobody is enrolled in.

```
select course
from class
where not exists
(select *
 from Takes
 where Takes.course = Class.course)
```

Derived relations

Find the CPR number of the student with the highest average grade

```
select cpr, avg(grade)
from Takes
group by cpr
having avg(grade) >= all
      (select avg(grade)
       from Takes
       group by cpr)
```

Both relations are pretty similar

Derived relations

Find the CPR number of the student with the highest average grade assuming we have pre computed an auxiliary table Aux(cpr,average)

```
select cpr, average
From Aux
where average =
      (select max(average)
       from Aux)
```

Aux

cpr	average
140298-1234	10
041297-5367	11
140789-7612	7
010292-3333	4

Derived relations

How to precompute Aux?

```
select cpr, avg(grade) as average
from Takes
group by cpr
```

Aux

cpr	average
140298-1234	10
041297-5367	11
140789-7612	7
010292-3333	4

Putting all together

Find the CPR number of the student with the highest average grade

```
select cpr, average
from (select cpr, avg(grade) as average
      from Takes
      group by cpr) as Aux
where average =
    (select max(average)
     from Aux)
```

Data Manipulation Language DML

- Select – From – Where
- Renaming
- Set operation
- Ordering
- Aggregates
- Nested subqueries
- Insertion
- Joins

Insertions

```
insert into Student
values (“010192-1329”,“Lone”,“Lyngby”)

insert into Student(ssn,name,address)
values (“031193-1339”,“Jens”,“Odense”)

insert into Student
select ssn, name, address
from Foreign-student
```

Deletions

```
delete from Student
where name="Lone"
```

Deletes all records where the name is “Lone”!

```
delete from Student
```

Deletes all records where the name is “Lone”!

Updates

Jesper got a 12 in SIDD.

```
update Takes
set    grade=12
where  name="Jesper"
and    course="SIDD"
```

Data Manipulation Language DML

- Select – From – Where
- Renaming
- Set operation
- Ordering
- Aggregates
- Nested subqueries
- Insertion
- Joins

Natural Join

Student

cpr	name	address
140298-1234	Jesper	Copenhagen
041297-5367	Nikoline	Aarhus
151197-2352	Claus	Dragør
050596-1142	Martin	Copenhagen

Takes

cpr	course	grade
140298-1234	SIDD	10
041297-5367	SIDD	12
280296-2222	CSYS	8

Student ⋈ Takes

cpr	name	address	course	grade
140298-1234	Jesper	Copenhagen	SIDD	10
041297-5367	Nikoline	Aarhus	SIDD	12

Natural Join

Verbose, but ok:

```
select cpr, name, address,
       course, grade
from Student Join Takes
     on Student.cpr = Takes.cpr
```

Natural Join

Verbose, but ok:

```
select cpr, name, address,
       course, grade
from Student Join Takes
     on Student.cpr = Takes.cpr
```

Also called: Inner Join

However, observe:

Student

cpr	name	address
140298-1234	Jesper	Copenhagen
041297-5367	Nikoline	Aarhus
151197-2352	Claus	Dragør
050596-1142	Martin	Copenhagen

Takes

cpr	course	grade
140298-1234	SIDD	10
041297-5367	SIDD	12
280296-2222	CSYS	8

Lost Information

Student ⋈ Takes

cpr	name	address	course	grade
140298-1234	Jesper	Copenhagen	SIDD	10
041297-5367	Nikoline	Aarhus	SIDD	12

Ways to Safe Information

- Outer join
- Left outer join

Outer Joins

Student

cpr	name	address
140298-1234	Jesper	Copenhagen
041297-5367	Nikoline	Aarhus
151197-2352	Claus	Dragør
050596-1142	Martin	Copenhagen

Takes

cpr	course	grade
140298-1234	SIDD	10
041297-5367	SIDD	12
280296-2222	CSYS	8

Student ⋈ Takes

cpr	name	address	course	grade
140298-1234	Jesper	Copenhagen	SIDD	10
041297-5367	Nikoline	Aarhus	SIDD	12
151197-2352	Claus	Dragør	NULL	NULL
050596-1142	Martin	Copenhagen	NULL	NULL
280296-2222	NULL	NULL	CSYS	8

Right Outer Join

Be Aware of Null Values

Null values are semantically ambigious!

$cond_1$	$cond_2$	$cond_1$ AND $cond_2$	$cond_1$ OR $cond_2$
true	*true*	*true*	*true*
true	*false*	*false*	*true*
true	*unknown*	*unknown*	*true*
false	*true*	*false*	*true*
false	*false*	*false*	*false*
false	*unknown*	*false*	*unknown*
unknown	*true*	*unknown*	*true*
unknown	*false*	*false*	*unknown*
unknown	*unknown*	*unknown*	*unknown*

cond	NOT *cond*
true	*false*
false	*true*
unknown	*unknown*

Conclusion

Complete guide to SQL as a

Data Manipulation Language (DML)

Next time

SQL as a Data Definition Language (DDL)