SQL 1: Basic Statements

Yufei Tao

Department of Computer Science and Engineering Chinese University of Hong Kong

SQL 1: Basic Statements

Structured query language (SQL) is a user-friendly language for specifying relational algebra queries. It is supported by all the major database systems. In this lecture, we will learn how to rewrite algebra operators in SQL.

select distinct $A_1, A_2, ..., A_n$ from $T_1, ..., T_m$ where P

where $T_1, ..., T_m$ are tables, $A_1, ..., A_n$ are attributes, and P is a predicate. The statement returns a table, and corresponds to the following relational algebra query:

 $\Pi_{A_1,\ldots,A_n}(\sigma_P(T_1\times\ldots\times T_m))$

select * from T where P

corresponds to

 $\sigma_P(T)$

æ

Э

□ ▶ ▲ 臣 ▶ ▲

pid	name	dept	rank	sal
p1	Adam	CS	asst	6000
p2	Bob	EE	asso	8000
<i>p</i> 3	Calvin	CS	full	10000
<i>p</i> 4	Dorothy	EE	asst	5000
<i>p</i> 5	Emily	EE	asso	8500
<i>p</i> 6	Frank	CS	full	9000

select * from PROF where rank = 'asst'

 $\sigma_{\mathrm{rank}} = "asst" (PROF)$

▶ < ≣ ▶ <</p>

pid	name	dept	rank	sal
<i>p</i> 1	Adam	CS	asst	6000
<i>p</i> 2	Bob	EE	asso	8000
<i>p</i> 3	Calvin	CS	full	10000
<i>p</i> 4	Dorothy	EE	asst	5000
<i>p</i> 5	Emily	EE	asso	8500
<i>p</i> 6	Frank	CS	full	9000

select *
from PROF
where not(rank = 'asst' and dept = 'EE')

 $\sigma_{\neg(\text{rank} = \text{``asst''} \land \text{dept} = \text{``EE''})}(\text{PROF})$

- ∢ ≣ ▶

=

select * from T where P

- In P, you can specify the standard comparisons and logic operators:
 - $\bullet \ =, <>, <, <=, >, >=$
 - Connect multiple comparisons with: AND, OR, NOT.

select distinct $A_1, ..., A_n$ from T

corresponds to

 $\Pi_{A_1,\ldots,A_n}(T)$

母▶ ∢ ≣▶

글▶ 글

pid	name	dept	rank	sal
<i>p</i> 1	Adam	CS	asst	6000
p2	Bob	EE	asso	8000
<i>p</i> 3	Calvin	CS	full	10000
<i>p</i> 4	Dorothy	EE	asst	5000
<i>p</i> 5	Emily	EE	asso	8500
<i>p</i> 6	Frank	CS	full	9000

select distinct dept, rank from PROF

$\Pi_{dept, rank}(PROF)$

Note

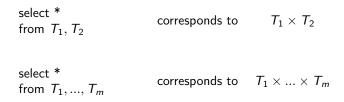
The keyword distinct removes all duplicate rows in the output. Omitting the keyword keeps all duplicates. See the next slide.

pid	name	dept	rank	sal
<i>p</i> 1	Adam	CS	asst	6000
p2	Bob	EE	asso	8000
<i>p</i> 3	Calvin	CS	full	10000
<i>p</i> 4	Dorothy	EE	asst	5000
<i>p</i> 5	Emily	EE	asso	8500
<i>p</i> 6	Frank	CS	full	9000

"select dept, rank from PROF" returns:

dept	rank
CS	asst
EE	asso
CS	full
EE	asst
EE	asso
CS	full

This duplicate-retaining feature is useful for aggregate queries as we will discuss later in the course.



TEACH

\mathbf{pid}	name	\mathbf{dept}	rank	sal
p1	Adam	CS	asst	6000
p2	Bob	EE	asso	8000
p3	Calvin	CS	full	10000
p4	Dorothy	EE	asst	5000
p5	Emily	EE	asso	8500

\mathbf{pid}	cid	year
p1	c1	2011
p2	c2	2012
p1	c2	2012

< ≣ >

э

э

select * from PROF, TEACH

 $\mathrm{PROF}\times\mathrm{TEACH}$

Putting Multiple Operators Together

PROF

TEA	CH
-----	----

\mathbf{pid}	name	\mathbf{dept}	rank	sal
p1	Adam	CS	asst	6000
p2	Bob	EE	asso	8000
p3	Calvin	CS	full	10000
p4	Dorothy	\mathbf{EE}	asst	5000
p5	Emily	EE	asso	8500

<u> </u>	pid	cid	year
<u> </u>	p1	<i>c</i> 1	2011
<u>,</u>	p2	<i>c</i> 2	2012
<u> </u>	p1	<i>c</i> 2	2012

同 ト イ ヨ ト イ ヨ ト

select distinct dept from PROF, TEACH where PROF.pid = TEACH.pid

 $\Pi_{dept}(\sigma_{PROF.pid} = TEACH.pid(PROF \times TEACH))$

select ... from T as S where ...

corresponds to

 $\dots \rho_S(T) \dots$

æ

-

白とくヨとく

TEACH

\mathbf{pid}	name	dept	rank	sal
p1	Adam	CS	asst	6000
p2	Bob	EE	asso	8000
p3	Calvin	CS	full	10000
p4	Dorothy	EE	asst	5000
p5	Emily	EE	asso	8500

\mathbf{pid}	cid	year
p1	c1	2011
p2	<i>c</i> 2	2012
p1	c2	2012

· < Ξ > <

select distinct dept from PROF as A, TEACH as B where A.pid = B.pid

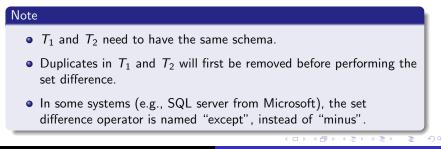
 $\Pi_{\rm dept}(\sigma_{\rm A.pid} = B.pid(\rho_{\rm A}({\rm PROF}) \times \rho_{\rm B}({\rm TEACH})))$

([SQL statement 1]) minus ([SQL statement 2])

corresponds to

$$T_1 - T_2$$

where $T_1(T_2)$ is the table returned by SQL statement 1 (2).



pid	name	dept	rank	sal
<i>p</i> 1	Adam	CS	asst	6000
p2	Bob	EE	asso	8000
<i>p</i> 3	Calvin	CS	full	10000
<i>p</i> 4	Dorothy	EE	asst	5000
<i>p</i> 5	Emily	EE	asso	8500
<i>p</i> 6	Frank	CS	full	9000

(select rank from PROF) minus (select rank from PROF where dept = 'CS')

 $\Pi_{\rm rank}({\rm PROF}) - \Pi_{\rm rank}(\sigma_{\rm dept\ =\ "CS"}({\rm PROF}))$

-∢ ≣ ▶

([SQL statement 1]) union ([SQL statement 2])

corresponds to

$T_1 \cup T_2$

where $T_1(T_2)$ is the table returned by SQL statement 1 (2).

Note

- T_1 and T_2 need to have the same schema.
- Duplicates in T_1 and T_2 will first be removed before performing the set union.

pid	name	dept	rank	sal
<i>p</i> 1	Adam	CS	asst	6000
<i>p</i> 2	Bob	EE	asso	8000
<i>p</i> 3	Calvin	CS	full	10000
<i>p</i> 4	Dorothy	EE	asst	5000
<i>p</i> 5	Emily	EE	asso	8500
<i>p</i> 6	Frank	CS	full	9000

(select * from PROF where sal <= 6000) union

(select * from PROF where sal >= 9000)

 $\sigma_{\rm sal} \leq 6000 ({\rm PROF}) \cup \sigma_{\rm sal} \geq 9000 ({\rm PROF})$

We have shown how to rewrite the 6 fundamental algebra operators in SQL. How about the extended operators $\leftarrow, \cap, \bowtie$ and \div ? As we will see next, there is an explicit statement only for \cap . Nevertheless, as \cap and \bowtie can be implemented using the 6 fundamental operators, they can also be written in SQL using the statements introduced earlier. We will, however, ignore \leftarrow from our discussion (this operator is the least useful one, anyway).

([SQL statement 1]) intersect ([SQL statement 2])

corresponds to

$T_1 \cap T_2$

where $T_1(T_2)$ is the table returned by SQL statement 1 (2).

Note

- T_1 and T_2 need to have the same schema.
- Duplicates in T_1 and T_2 will first be removed before performing the set union.

pid	name	dept	rank	sal
<i>p</i> 1	Adam	CS	asst	6000
<i>p</i> 2	Bob	EE	asso	8000
<i>p</i> 3	Calvin	CS	full	10000
<i>p</i> 4	Dorothy	EE	asst	5000
<i>p</i> 5	Emily	EE	asso	8500
<i>p</i> 6	Frank	CS	full	9000

(select * from PROF where sal >= 6000) intersect

(select * from PROF where dept = 'CS')

 $\sigma_{\mathrm{sal} \ge 6000}(\mathrm{PROF}) \cap \sigma_{\mathrm{dept} = \mathrm{"CS"}}(\mathrm{PROF})$

TEACH

\mathbf{pid}	name	\mathbf{dept}	rank	sal
p1	Adam	CS	asst	6000
p2	Bob	EE	asso	8000
p3	Calvin	CS	full	10000
p4	Dorothy	EE	asst	5000
p5	Emily	EE	asso	8500

\mathbf{pid}	cid	year
p1	c1	2011
p2	c2	2012
p1	c2	2012

select distinct A.pid, name, dept, rank, sal, cid, year from PROF, TEACH where PROF.pid = TEACH.pid

 $\Pi_{\text{PROF.pid, name, dept, rank, sal, cid, year}}(\sigma_{\text{PROF.pid} = \text{TEACH.pid}}(\text{PROF} \times \text{TEACH}))$

=

 $\operatorname{PROF}\bowtie\operatorname{TEACH}$

Division

 $\begin{array}{c} T_2 \\ \hline cid \\ \hline c1 \\ \hline c2 \\ \hline c3 \end{array}$

(select pid from T_1) minus select pid from ((select * from (select pid from T_1), T_2) minus (select * from T_1))

Note

Notice how an SQL statement can be nested in a from clause.

⊒ ▶

$$\Pi_{S_1-S_2}(T_1) - \Pi_{S_1-S_2} \Big(\Pi_{S_1-S_2}(T_1) \times T_2 - T_1 \Big) = T_1 \div T_2$$