

CMPT 354 - Database Systems & Structures - Official Query Language Crib Sheet

You will be given a copy of this with the course exams.

Relational algebra: Duplicates are eliminated, general expressions are formed out of smaller subexpressions using:

- $\sigma_p(E_1)$ select (p a predicate)
- $\Pi_s(E_1)$ project (s a list of attributes)
- $\rho_x(E_1)$ rename (x a relation name)
- $E_1 \cup E_2$ union
- $E_1 - E_2$ set difference
- $E_1 \times E_2$ cartesian product
- $E_1 \bowtie E_2$ natural join
- $E_1 \cap E_2$ intersect
- $E_1 \div E_2$ division
- $E_1 \leftarrow E_2$ assignment
- $\delta_{A \leftarrow E}(r)$ update

SQL: Typical query has the form:

```
select  $A_1, A_2, \dots, A_n$ 
from  $r_1, r_2, \dots, r_m$ 
where  $P$ 
```

where each A_i represents an attribute, each r_i a relation, and P is a predicate.

- SQL allows duplicates.
- **distinct:** remove duplicates
- set operations **union**, **intersect** and **minus**
- predicates can involve
 - logical connectives **and**, **or** and **not**.
 - arithmetic expressions on constant or tuple values.
 - **between** operator for ranges of values.
 - string matching operators **%** (any substring) and **_** (underscore, matching any character).
 - **like** operator to use string matching.
 - **not like** for string mismatching.
- **in** and **not in** operations for set membership.
- equality or inequality operators with **some** or **all**.
- **contains:** set containment

- **order by** makes tuples appear in sorted order
- **desc** specifies descending order
- **asc** specifies ascending order
- average value — **avg**
- minimum value — **min**
- maximum value — **max**
- total sum of values — **sum**
- number in group — **count**
- tuples satisfying **where** clause are placed into groups by the **group by** clause
- the **having** clause is applied to each group
- groups satisfying the having clause are used by the **select** clause to generate the result tuples
- if no **having** clause is present, the tuples satisfying the **where** clause are treated as a single group
- **delete:** delete tuples
- **insert into:** insert tuples
- **update:** update tuples
- **create view v as:** create view
- **create table:** create a relation
- **drop table:** destroy a relation
- **alter table:** change a relation

QBE: Uses skeleton tables.

- Domain variables are preceded by an underscore character.
- Constants appear without any qualification.
- A P. in variable causes printing.
- A P.ALL. prefix suppresses duplicate elimination.
- A P. in front of the row prints all attributes.
- The domain variable may be omitted if it is not used elsewhere.
- Arithmetic expressions are allowed.
- Comparison operators are allowed, space on left hand side is left blank.
- Queries on several relations require several skeleton tables.
- **condition box** may be used.
- Logical expressions $\vee$, $\wedge$ and $\neg$.
- Temporary *result* relation can be created.

- The order in which tuples are displayed can be controlled by adding the command AO. (ascending order) or DO. (descending order) to the print command.
- QBE includes the aggregate operators AVG, MAX, MIN, SUM and CNT. As QBE eliminates duplicates by default, they must have ALL. appended to them.
- To override the ALL. we can add UNQ. (unique).
- To compute functions on groups, we use the G. operator.
- Delete: use D. instead of the P. operator. Whole tuples may be deleted, or only some columns.
- Insertion uses the I. operator.
- We can update individual attributes with the U. operator. Fields left blank are not changed.

Sample query:

<i>borrow</i>	<i>bname</i>	<i>loan#</i>	<i>cname</i>	<i>amount</i>
	Perryridge		└x	

<i>customer</i>	<i>cname</i>	<i>street</i>	<i>ccity</i>
	P.└x		P.└y

Domain Relational Calculus:

An expression is of the form

$$\{ \langle x_1, x_2, \dots, x_n \rangle \mid P(x_1, x_2, \dots, x_n) \}$$

where the $x_i, 1 \leq i \leq n$, represent domain variables, and P is a **formula**.

Logical connectives $\vee, \wedge$ and $\neg$, plus existential and universal quantifiers ($\exists$ and $\forall$) are allowed. Implies ($\Rightarrow$) also used (If P , then Q).

A sample query:

$$\begin{aligned} & \{ \langle c, x \rangle \mid \exists b, l, a (\langle b, l, c, a \rangle \in \text{borrow} \\ & \wedge b = \text{"Perryridge"} \\ & \wedge \exists y (\langle c, y, x \rangle \in \text{customer})) \} \end{aligned}$$

Tuple relational calculus:

Nonprocedural language. A query in the tuple relational calculus is expressed as

$$\{ t \mid P(t) \}$$

i.e. the set of tuples t for which predicate P is true.

Logical connectives $\vee, \wedge$ and $\neg$, plus existential and universal quantifiers ($\exists$ and $\forall$) are allowed. Implies ($\Rightarrow$) also used (If P , then Q).

A sample query:

$$\begin{aligned} & \{ t \mid \exists s \in \text{borrow} (t[\text{cname}] = s[\text{cname}] \\ & \wedge s[\text{bname}] = \text{"Perryridge"}) \\ & \vee \exists u \in \text{deposit} (t[\text{cname}] = u[\text{cname}] \\ & \wedge u[\text{bname}] = \text{"Perryridge"}) \} \end{aligned}$$