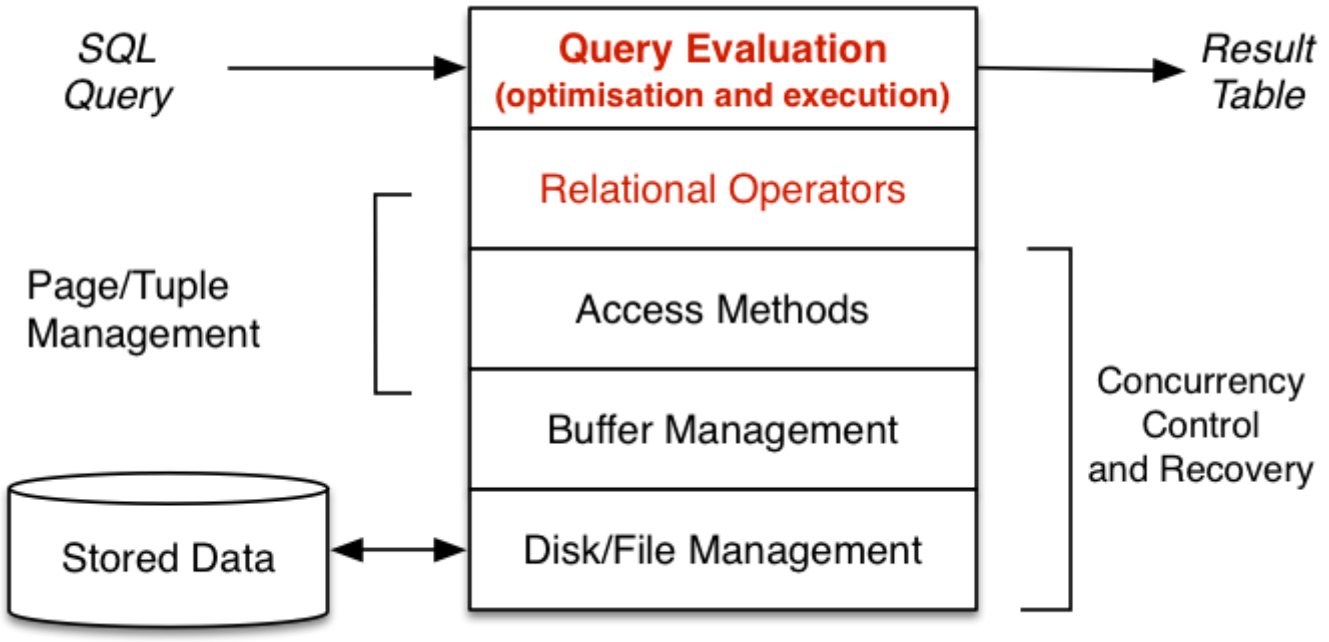
>>

Query Processing

- Query Processing
- Terminology Variations

^ >>

❖ Query Processing



<< ^ >>

❖ Query Processing (cont)

A query in SQL:

- states *what* kind of answers are required (declarative)
- does not say *how* they should be computed (procedural)

A query evaluator/processor :

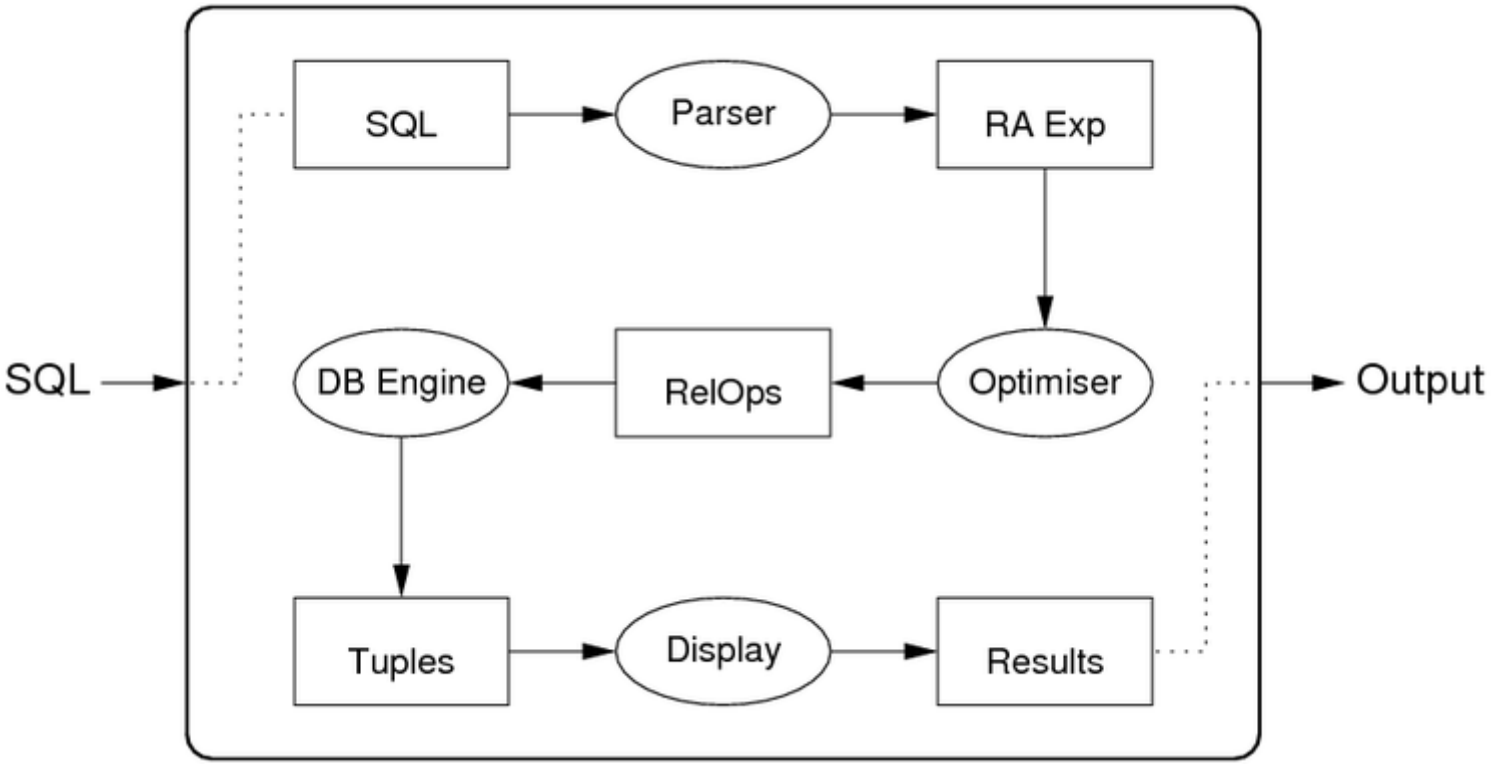
- takes declarative description of query (in SQL)
- parses query to internal representation (relational algebra)
- determines plan for answering query (expressed as DBMS ops)
- executes method via DBMS engine (to produce result tuples)

Some DBMSs can save query plans for later re-use.

<< ^ >>

❖ Query Processing (cont)

Internals of the query evaluation "black-box":



<< ^ >>

❖ Query Processing (cont)

DBMSs provide several "flavours" of each RA operation.

For example:

- several "versions" of selection (σ) are available
- each version is effective for a particular kind of selection, e.g

```
select * from R where id = 100   -- hashing
select * from S                  -- Btree index
where age > 18 and age < 35
select * from T                  -- MALH file
where a = 1 and b = 'a' and c = 1.4
```

Similarly, π and $\bowtie$ have versions to match specific query types.

<< ^ >>

❖ Query Processing (cont)

We call these specialised version of RA operations RelOps.

One major task of the query processor:

- given a RA expression to be evaluated
- find a combination of RelOps to do this efficiently

Requires the query translator/optimiser to consider

- information about relations (e.g. sizes, primary keys, ...)
- information about operations (e.g. selection reduces size)

RelOps are realised at execution time

- as a collection of inter-communicating nodes
- communicating either via pipelines or temporary relations

<< ^

❖ Terminology Variations

Relational algebra expression of SQL query

- intermediate query representation
- logical query plan

Execution plan as collection of RelOps

- query evaluation plan
- query execution plan
- physical query plan

Representation of RA operators and expressions

- σ = *Select* = *Sel*, π = *Project* = *Proj*
- $R \bowtie S$ = *R Join S* = *Join(R,S)*, $\wedge$ = &, $\vee$ = |

Produced: 5 Apr 2021