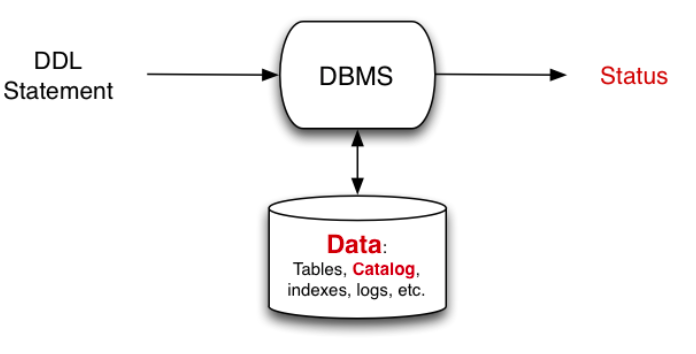
**Lecture 1**

**Data Definition Language (DDL)**

Relational data: relations/tables, tuples, values, types, adds meta-data to the database.

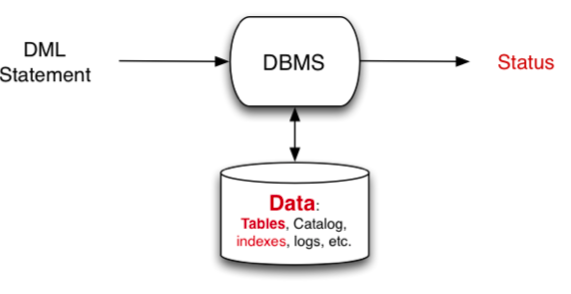
DBMSs typically store meta-data as special tables (catalog)

i.e. input – DDL statements; result – meta-data in catalog is modified



**Data Modification Language (DML)**

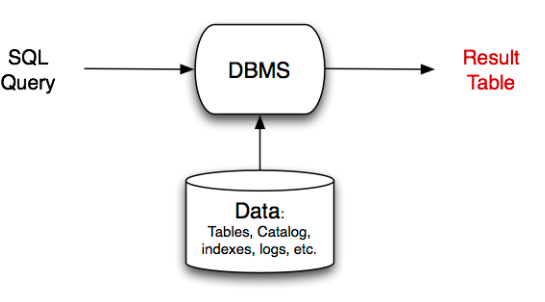
Changing data: insert, delete, update

i.e. input – DML statements; result – tuples are added, removed or modified

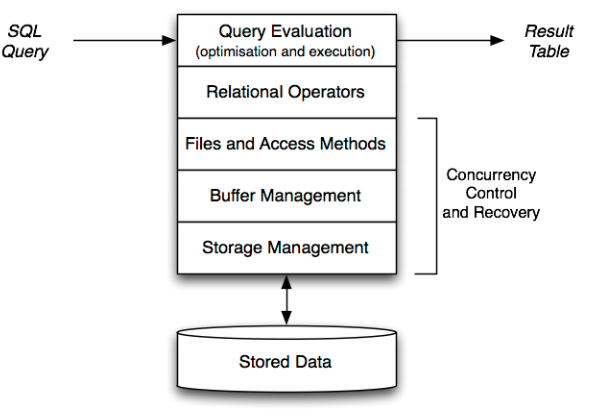
**Query Evaluator**

Read an SQL query and return a table giving result of query

i.e. input – SQL query; output – table (displayed as text)



**DBMS Architecture**



Important factors related to DBMS architecture

* Data is stored permanently on large slow devices
* Data is processed in small fast memory

Implications:

* Data structures should minimise storage utilisation
* Algorithms should minimise memory/disk data transfers

Modern DBMSs interact with storage via the O/S file-system

**Database Engine Operations**

DB engine = “relational algebra virtual machine”

Operators: selection (σ), projection (π), join (⎢×⎢), union (∪), intersection (∩), difference (-)

Extensions: sort (order by), partition (group by), aggregation

**Relational Algebra (RA)**

Mathematical system for manipulating relations, or DML for relational model (RM)

**Computational view of RA operations:**

**Selection**:

|  |
| --- |
| result = {}  for each tuple t in relation r  If (C(t) {result = result U {t}} // C is Boolean function that tests selection condition |

**Projection**:

|  |
| --- |
| result = {}  for each tuple t in relation r  result = result U {t[X]} // result schema is given by attributes in X |

**Union**:

|  |
| --- |
| result = r1  for each tuple t in relation r2  result = result U {t} |

**Intersection**:

|  |
| --- |
| result = {}  for each tuple t in relation r1  if (t ∈ r2) {result = result U {t}} |

**Theta Join**:

|  |
| --- |
| result = {}  for each tuple t1 in relation r  for each tuple t2 in relation s  if (matches (t1, t2, C)) // C is the join condition  result = result U {concat (t1, t2)} |

**Left Outer Join**:

|  |
| --- |
| result = {}  for each tuple t1 in relation r  nmatches = 0  for each tuple t2 in relation s  if (matches (t1, t2, C))  result = result U {combine (t1, t2)}  nmatches++  if (nmatches == 0)  // Snull is a tuple with schema S and all attributes set to NULL  result = result U {combine (t1, Snull)} |

**PostgreSQL Functionality**

Uses multi-version concurrency control (**MVCC**)

* Multiple “versions” of the database exist together
* A transaction sees the version that was valid at its start time
* Readers don’t block writers; writers don’t block readers
* This significantly reduces the need for locking

Disadvantages of this approach:

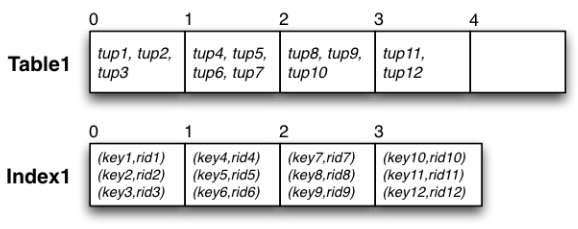
* Extra storage for old versions of tuples (**vacuum** fixes this)

PostgreSQL has a well-defined and open extensibility model

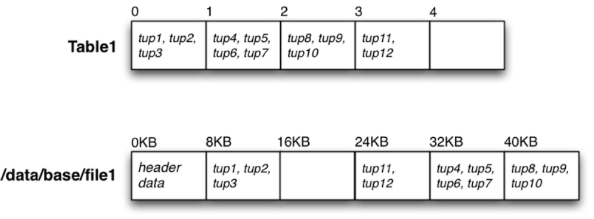
* Stored procedures are held in database as strings
* Can add new data types, operators, aggregates, indexes

**View of Data**

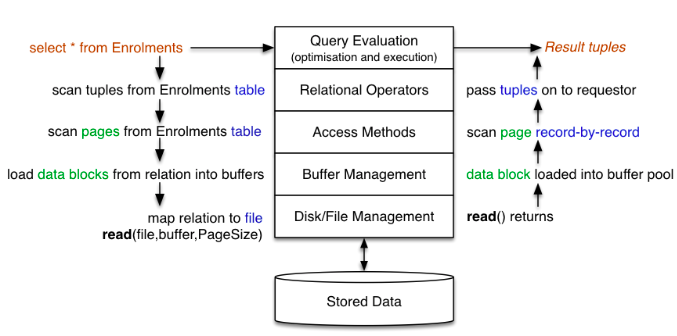
* Sequence of fixed-size pages, typically 1KB to 8KB
* Where each page contains tuple data or index data



* Maps (tableName, pageIndex) to (file, offset)



**Storage Manager Interface**

****

Examples of references (addresses) used in DBMSs:

**PageID = FileID + Offset** … identifies (locates) a block of data, where **Offset** gives location of block within file

**TupleiD = PageID + Offset** … identifies (locates) a single tuple, where **Offset** gives location of tuple within page

**Database Objects**

pg\_catalog – a global schema

pg\_database – contains information about databases

pg\_namespace – contains information about schemata

pg\_tablespace – contains information about tablespaces

pg\_class – holds core information about tables

pg\_attribute – contains information about attributes

pg\_type – contains information about types

* Information the RDBMS needs about relations is stored in the system catalog tables (SQL standard **information\_schema**)
* PostgreSQL tuples contain
  + Owner-specified attributes (from create table)
  + System-defined attributes