COL 362 & COL 632

Processing SQL

01 Feb 2023

Few other important features in SQL

- Indexes
- Views
- Authorization / ACL

- Data structure to access specific tuples “fast”
- Very important for query processing and query optimization
- Useful when no. of results *very small* compared to the total no. of tuples

```
SELECT * FROM Students
WHERE Name = ‘Rana Prathap’;
```

Likely to be just one result ☺

```
CREATE INDEX NameIndex
ON Actor (Name)
```

Few other important features in SQL

- Indexes
- Views
- Authorization / ACL

- Views are tables created from existing tables
  - Materialized (physically exist)
    - `CREATE MATERIALIZED VIEW`
  - Virtual (don't physically exist)
    - `CREATE VIEW`
- Offer a "simplified" view of the data
  - **Secure** data from non-authorized users

```
CREATE VIEW Teachers4Students AS
SELECT F_NAME, L_NAME, AGE, OFFICE
FROM Teachers WHERE DEPT = 'CSE';
--- we have not retrieved their salary
```

- Materialized views used in speeding up query processing

- How to update views when base tables are updated?
- How to propagate changes when views are updated?

Few other important features in SQL

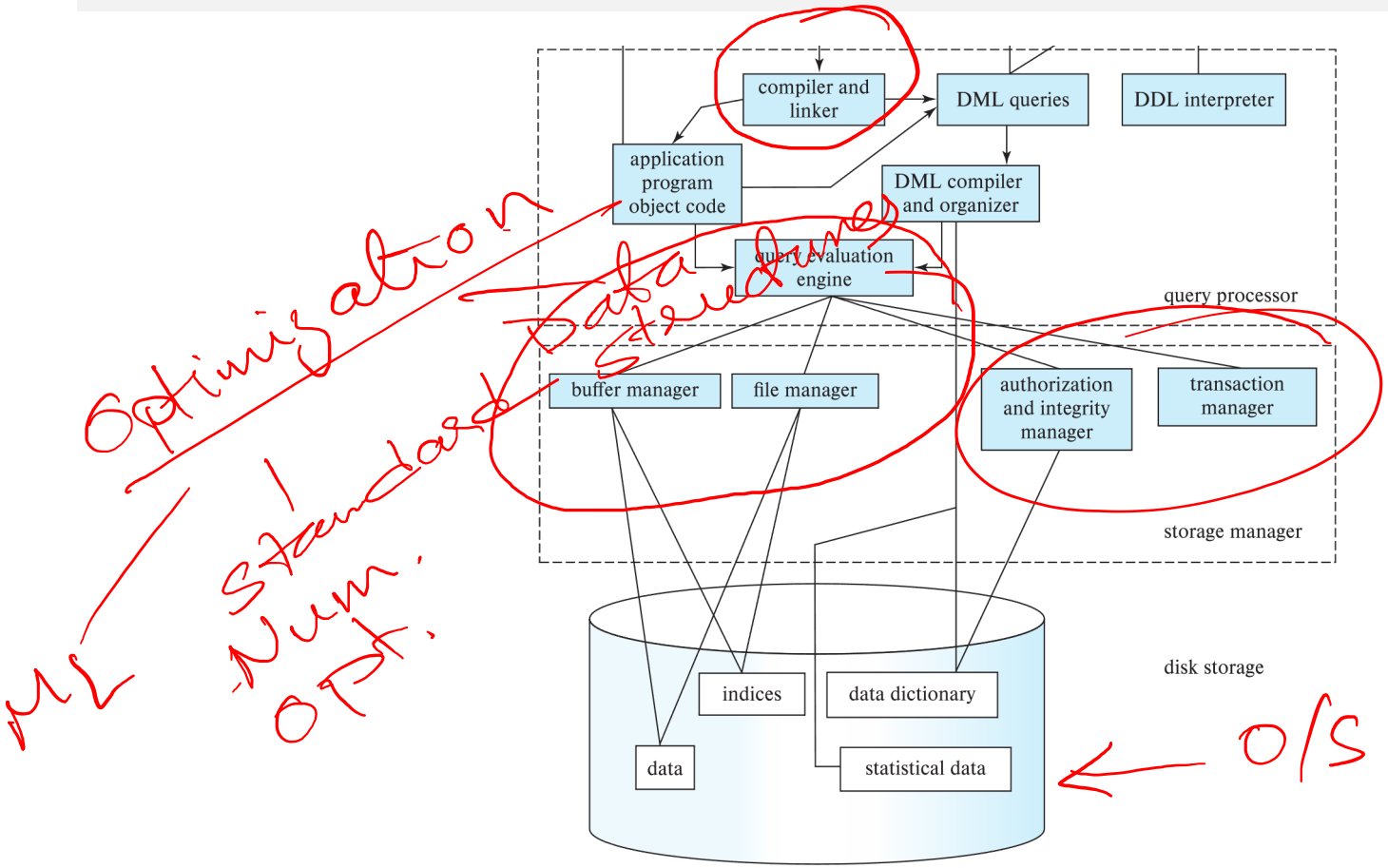
- Indexes
- Views
- Authorization / ACL

- Nearly all database systems provide authorization and access control mechanisms for the databases
- Users: individual ”login”s to the database system
- Roles: functional roles defined by the DBA and users are assigned to these roles

Database Engine – or a DBMS

- A database system is partitioned into modules that deal with each of the responsibilities of the overall system.
- The functional components of a database system can be divided into
  - The **storage manager**,
  - The **query processor** component,
  - The **transaction management** component.

DBMS Architecture (Centralized/Shared-Memory)

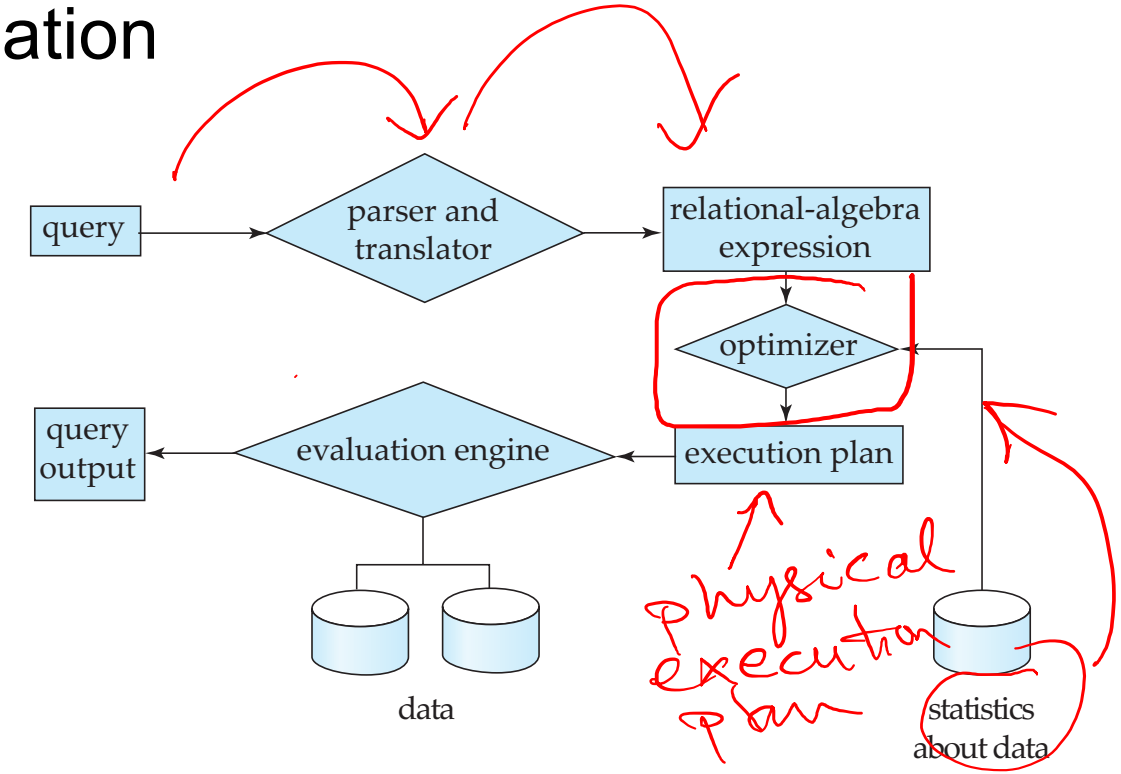


Query Processor

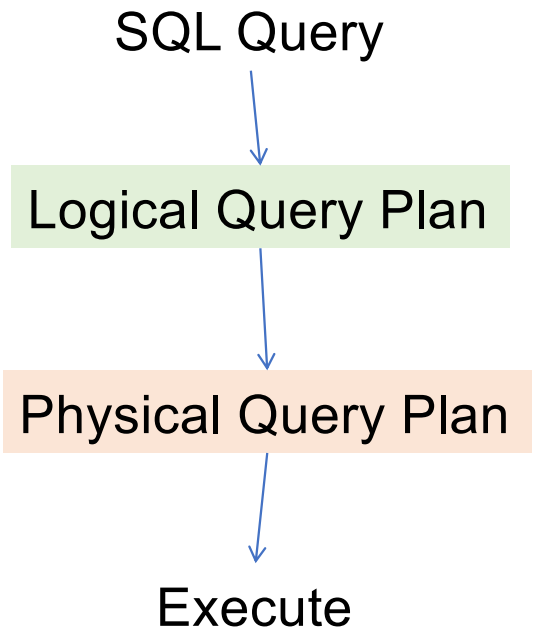
- The query processor components include:
  - DDL interpreter -- interprets DDL statements and records the definitions in the data dictionary.
  - DML compiler -- translates DML statements in a query language into an evaluation plan consisting of low-level instructions that the query evaluation engine understands.
    - The DML compiler performs query optimization; that is, it picks the lowest cost evaluation plan from among the various alternatives.
  - Query evaluation engine -- executes low-level instructions generated by the DML compiler.

Query Processing

1. Parsing and translation
2. Optimization
3. Evaluation



Logical and Physical Query Plans



- A relational algebra expression may have many equivalent expressions
  - $\sigma_{salary<75000}(\Pi_{salary}(instructor)) == \Pi_{salary}(\sigma_{salary<75000}(instructor))$
- Several different algorithms for each operator
  - Correspondingly, a relational-algebra expression can be evaluated in many ways.
- **evaluation-plan:** Annotated expression specifying detailed evaluation strategy
  - Use an index on *salary* to find instructors with salary < 75000,
  - Or perform complete relation scan and discard instructors with salary ≥ 75000

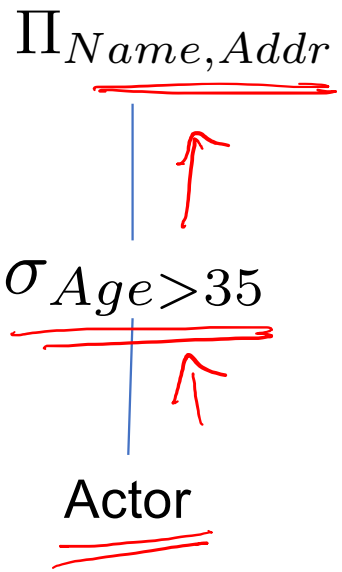
Plans and Equivalent Plans

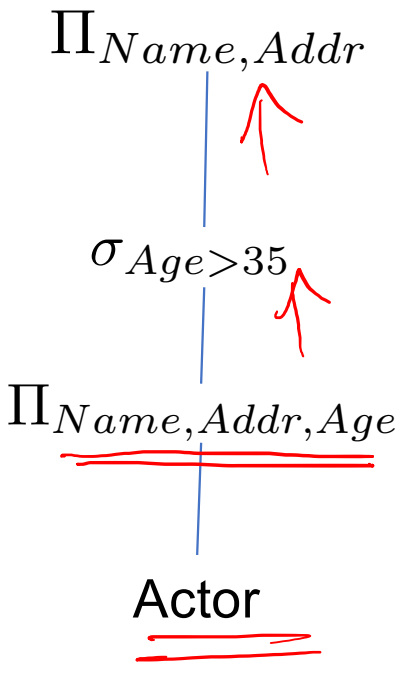
Return the names and addresses of actors over 35

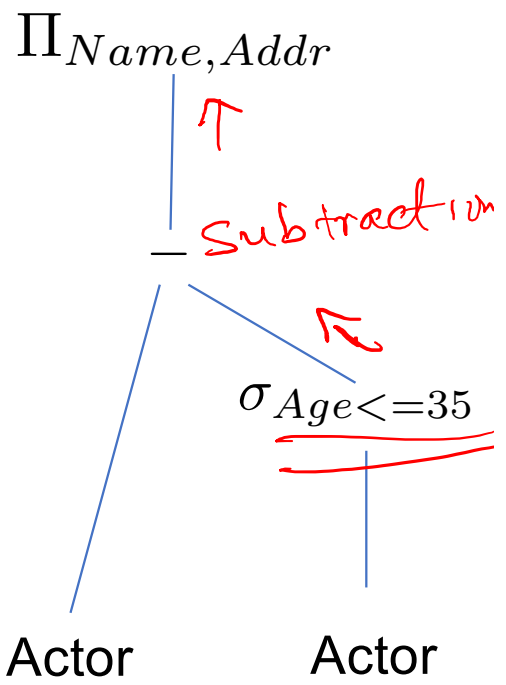
$$\Pi_{Name,Addr}(\sigma_{Age>35}(Actor))$$

$$R1 = \sigma_{Age>35}(Actor)$$

$$R2 = \Pi_{Name,Addr}(R1)$$







Algebraic Laws

Commutativity and Associativity

$$R \times S = S \times R$$

$$(R \times S) \times T = R \times (S \times T)$$

$$R \bowtie S = S \bowtie R;$$

$$(R \bowtie S) \bowtie T = R \bowtie (S \bowtie T)$$

$$\sigma_{C1 \text{ AND } C2}(S) = \sigma_{C1}(\sigma_{C2}(S))$$

$$\sigma_{C1 \text{ AND } C2}(S) = \sigma_{C1}(S) \cap \sigma_{C2}(S)$$

$$\sigma_{C1 \text{ OR } C2}(S) = \sigma_{C1}(S) \cup \sigma_{C2}(S)$$

$$\sigma_{C1}(\sigma_{C2}(S)) = \sigma_{C2}(\sigma_{C1}(S))$$

$$\sigma_C(S \cup T) = \sigma_C(S) \cup \sigma_C(T)$$

$$\sigma_C(S \bowtie T) = \sigma_C(S) \bowtie T$$

$$\sigma_C(S \bowtie T) = \sigma_C(S) \bowtie \sigma_C(T)$$

Example (1/5)

Actors

Name	Age	Addr
Priyanka Chopra	36	Mumbai
Anthony Hopkins	81	LA
Bill Nighy	69	LA
Abhishek Bachchan	42	Mumbai

Movies

Name	Year	Title
Priyanka Chopra	2011	Don-II
Anthony Hopkins	2011	MI-IV
Bill Nighy	2009	Valkyrie
Abhishek Bachchan	2010	Raavan

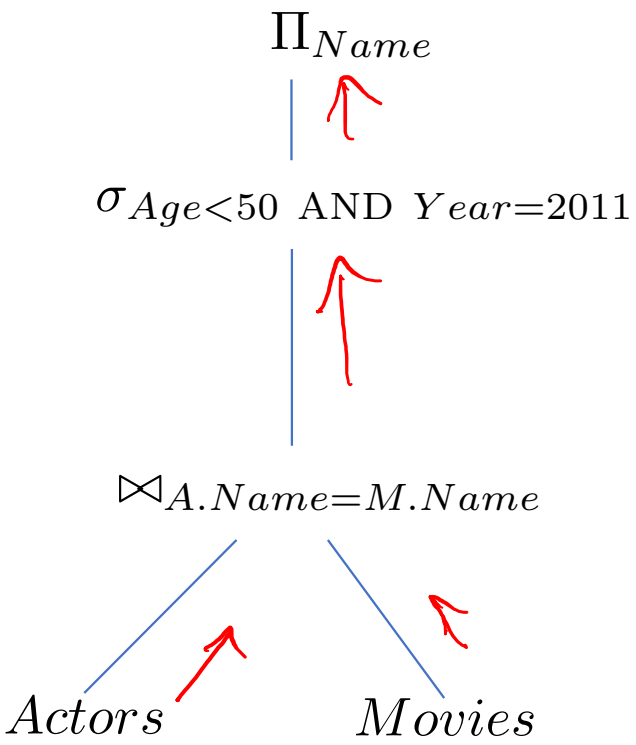
Return the names of actors below the age of 50 who have acted in a movie in 2011

$$\Pi_{Name}(\sigma_{Age<50 \text{ AND } Year=2011}(Actors \bowtie_{A.Name=M.Name} Movies))$$

$$\begin{aligned} Allmovies &= Actors \bowtie_{A.Name=M.Name} Movies \\ Movies1 &= \sigma_{Age<50 \text{ AND } Year=2011}(AllMovies) \\ Result &= \Pi_{Name}(Movies1) \end{aligned}$$

Example (2/5)

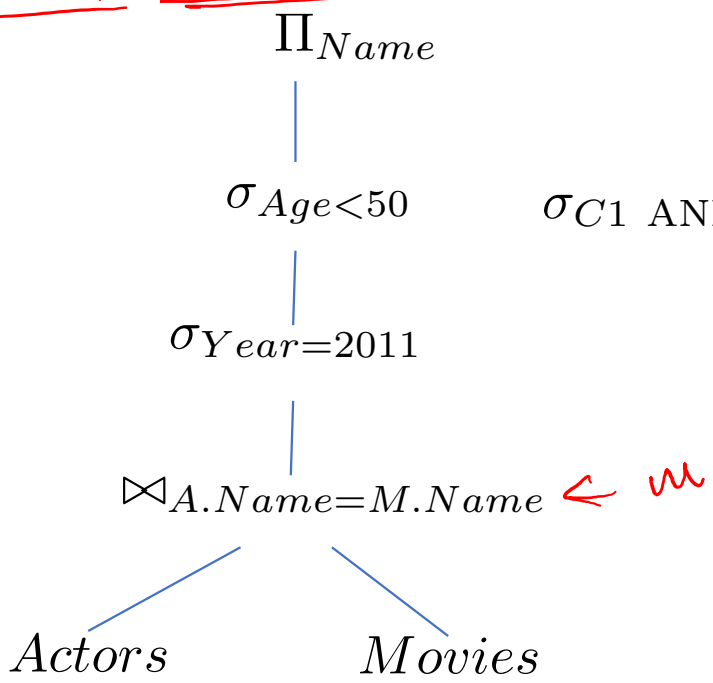
$$\Pi_{Name}(\sigma_{Age<50 \text{ AND } Year=2011}(Actors \bowtie_{A.Name=M.Name} Movies))$$



Example (3/5)

$$\Pi_{Name}(\sigma_{Age<50 \text{ AND } Year=2011}(Actors \bowtie_{A.Name=M.Name} Movies))$$

$$\Pi_{Name}(\sigma_{Age<50}(\sigma_{Year=2011}(Actors \bowtie_{A.Name=M.Name} Movies)))$$



$$\sigma_{C1 \text{ AND } C2}(S) = \sigma_{C1}(\sigma_{C2}(S))$$