Assignment Project Exam Help

https://eduassistpro.github.i

Add WeChat edu_assist_pr

Query Processing – Overview

1. Users **submit** SQL queries to a DBMS.
2. The DBMS **processes and executes** them in a database.

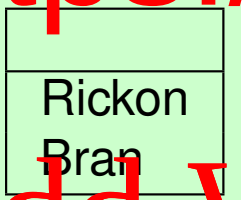
- **Note:** SQL is a declarative language, so it is the task of DBMSs to decide how SQL queries should be executed.

Query Processing – Example

- Steps:

```
S
```

- T



- **Questions:**
  - **How does a relational DBMS process this?**
  - **How can a relational DBMS process this efficiently?**

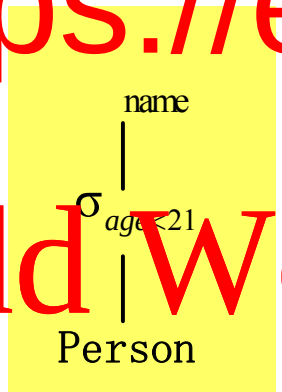
Query Processing – Example

```
SELECT name FROM Person WHERE age<21
```

High-level language (SQL)

⇓

Low-level language (relational algebra)



⇓

name
Rickon
Bran

Query result

Query Processing – Example

Query Processing Steps

- **Query parser and translator**
  1. Check the syntax of SQL queries
  2.
  3.

- **Query optimiser**
  1. Transform into the best possible executio
  2. Specify the implementation of each opera

- **Evaluation engine**
  1. Evaluate the query execution plan
  2. Return the result to the user

Query Processing – Parser

- The **parser** checks the syntax of the query:
  - Validation of table names, attributes, data types, access permission ...;
  - Either the query is executable or an error message is generated.

Query Processing – Parser

- Consider the relation schema:

  `Person(id:integer, name:string, age:integer, address:string)`

- **N** ge,
  w
  E

–	–		
id	Person		
name	Person		
age	Person		
address	Person		
...	...	...	...

- **Question:** Can the following query be accepted by the parser?

```
SELECT fname, lname FROM Person WHERE address<21;
```

Query Processing – Parser

- Consider the relation schema:

g)

- Q https://eduassistpro.github.i

```
SELECT fname, lname FROM Person WHERE ad
```


- **Answer:** The query **would be rejected**
  1. The attributes `fname` and `lname` are
  2. The attribute `address` is not comparable with 21.

Query Processing – Translator

- The **translator** translates queries into RA expressions (not necessarily equivalent due to duplicates):
  - A query is first decomposed into **query blocks**.
  -

Recall: RA and SQL Queries

- **RA operators**
  - **selection** σ_φ
  - **projection** $\pi_{A_1,\ldots,A_n}$
  - **difference** $R_1 - R_2$
- **SQL statement**

```
SELECT attribute_list
  FROM table_list
                ]
                  ist
                dition]]
                  ist];
```

$\sigma_\varphi(R) \Leftrightarrow$ SELECT F

$\pi_{A_1,\ldots,A_n}(R) \Leftrightarrow$ SELECT DISTIN

$R_1 \times R_2 \Leftrightarrow$ SELECT DISTINCT * FROM R_1, R_2;

...

- Aggregate operations in SQL require extended RA expressions.

Recall: RA and SQL Queries

- Nested subqueries are decomposed into separate query blocks.
- **Example:**

```
SELECT Lname, Fname
```


Outer query block

```
SELECT Lname, Fname FROM EMPLOYEE WHERE
Salary > c
```

⇓ translated

$\pi_{Lname, Fname}(\sigma_{Salary > c}(\text{EMPLOYEE}))$

```
ssn=5)
```

⇓ translated

$\pi_{Salary}(\sigma_{ssn=5}(\text{EMPLOYEE}))$

Query Processing – Query Optimiser

1. Transform into the best possible execution plan

(will be covered in this course)

2. Specify the implementation of each operator in th

There are different possible implementatio operator!

(will not be covered in this course)

Query Processing – Query Optimiser

- SQL queries only specify **what data to be retrieved** and **not how to retrieve data**.
- T
- Q **n plan**:
  1. enumerating alternative plans (typically
  2. choosing the one with the least estimated co
- Query optimisation is one of the most important tasks of a relational DBMS.
  A good DBMS must have a good query optimiser!

Equivalent RA Expressions

- Suppose that we have:

  St
  Ex
  Co


```
SELECT lastName, result, title
  FROM Students, Exams, Cou
 WHERE Students.matNr=Exams
       Exams.crsNr=Courses.
```


- **Question:**

  How many equivalent RA expressions for this SQL query can you find?

Equivalent RA Expressions


```
Students(matNr, firstName, lastName, email)
Exams(matNr, crsNr, result, semester)
Courses(crsNr, title, unit)
```



```
EXAMS.crsNr=COURSES.crsNr AND result  1.3;
```

- **Answer:**
  1. $\pi_{lastName,result,title}(\sigma_{result \leq 1.3}((\texttt{Student}$ ams) $\bowtie_{\sigma_{\texttt{Exams}.crsNr=\texttt{Courses}.crsNr}} \texttt{Courses}))$
  2. $\pi_{lastName,result,title}(\sigma_{result \leq 1.3}(\sigma_{\texttt{EXAMS}.crsNr=\texttt{Courses}.crsNr}(\sigma_{\texttt{Students}.matNr=\texttt{Exams}.matNr}(\texttt{Students} \times \texttt{Exams} \times \texttt{Courses}))))$
  3. $\pi_{lastName,result,title}((\texttt{Students} \bowtie_{\texttt{Students}.matNr=\texttt{Exams}.matNr}(\sigma_{result \leq 1.3}(\texttt{Exams}))) \bowtie_{\sigma_{\texttt{Exams}.crsNr=\texttt{Courses}.crsNr}} \texttt{Courses})$

Query Trees

- Each RA expression can be represented as a **query tree**:
  - **leaf nodes** represent the input relations;
  - **internal nodes** represent the intermediate result;
  -
- **E**

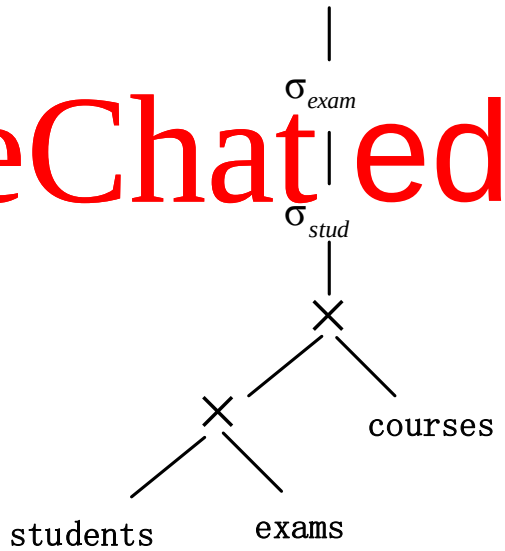
π_{l} ns)

▷

Query Trees

- **Exercise:** Can you draw the query tree for the following RA expression?

$$\pi_{lastName,result,title}(\sigma_{result \leq 1.3}(\sigma_{\texttt{Exams.crsNr=Courses.crsNr}}(\sigma_{\texttt{Students.matNr=Exams.matNr}}(\texttt{Students} \quad \texttt{Exams} \quad \texttt{Courses}))))$$



Query Trees

- For each query tree, computation proceeds **bottom-up**:
  - child nodes must be executed before their parent nodes;
  -

Execution Plan

- A **query execution plan** consists of an (extended) query tree with additional annotation at each node indicating:
  - (1) the *access method* to use for each table, and
  - (

Query Processing – Evaluation Engine

- The **evaluation engine** executes an execution plan, and returns the query answer to the user.