Australian National University

Week 8 – Query Processing and Optimisation

Housekeeping

1. Assignment 2 (Database Theory) for both COMP2400/6240 students:
   - The submission deadline is 23:59, Oct 11, 2022.
   - This assignment must be done individually (no group work).

Housekeeping

1. Assignment 2 (Database Theory) for both COMP2400/6240 students:
   - The submission deadline is 23:59, Oct 11, 2022.
   - This assignment must be done individually (no group work).
2. All the labs on Oct 3 (Monday, public holiday) in Week 9 will be moved to the same venues on Oct 10 (Monday) in Week 10.

Housekeeping

1. Assignment 2 (Database Theory) for both COMP2400/6240 students:
    - The submission deadline is 23:59, Oct 11, 2022.
    - This assignment must be done individually (no group work).

2. All the labs on Oct 3 (Monday, public holiday) in Week 9 will be moved to the same venues on Oct 10 (Monday) in Week 10.
3. Lab 8 is optional (no associated with any assessment items)
    - We will open a separate sign-up page on Wattle at 12pm Oct 6.
    - All the optional labs will be scheduled from Oct 11 to Oct 14.
    - Four options are available

      (1) Database Programming with Java

      (2) Database Programming with Python

      (3) Database Exercises on IMDB

      (4) Database Security (SQL Injection)

Query Processing – Example


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

High-level language
(SQL)

$\Downarrow$

$\pi_{name}(\sigma_{age<21}(\texttt{Person}))$

Low-level language
(Relational Algebra)

$\Downarrow$

π_{name}
|
$\sigma_{age<21}$
|
Person

Execution plan
(Query tree)

$\Downarrow$

name
Rickon
Bran

Query result

From SQL to RA Expressions

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

```
SELECT lastName, result, title
  FROM STUDENTS, EXAMS, COURSES
 WHERE STUDENTS.matNr=EXAMS.matNr AND
       EXAMS.crsNr=COURSES.crsNr AND result≤1.3;
```

From SQL to RA Expressions


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



```
SELECT lastName, result, title
  FROM STUDENTS, EXAMS, COURSES
 WHERE STUDENTS.matNr=EXAMS.matNr AND
       EXAMS.crsNr=COURSES.crsNr AND result≤1.3;
```

- **RA Expressions:**

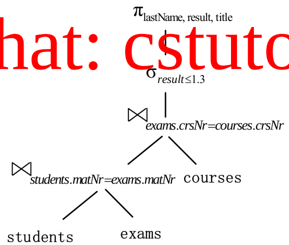
1. $\pi_{lastName,result,title}(\sigma_{result\leq 1.3}((\texttt{Students} \bowtie_{\texttt{Students}.matNr=\texttt{Exams}.matNr} \texttt{Exams}) \bowtie \sigma_{\texttt{Exams}.crsNr=\texttt{Courses}.crsNr}\ \texttt{Courses}))$
2. $\pi_{lastName,result,title}(\sigma_{result\leq 1.3}(\sigma_{\text{EXAMS}.crsNr=\texttt{Courses}.crsNr}(\sigma_{\texttt{Students.matNr=Exams.matNr}}(\texttt{Students} \times \texttt{Exams} \times \texttt{Courses}))))$
3. $\pi_{lastName,result,title}((\texttt{Students} \bowtie_{\texttt{Students}.matNr=\texttt{Exams}.matNr} (\sigma_{result\leq 1.3}(\texttt{Exams}))) \bowtie_{\texttt{Exams}.crsNr=\texttt{Courses}.crsNr} \texttt{Courses})$
4. ...

Australian National University

From RA Expressions to Query Trees

- Each RA expression can be represented as a **query tree**:
  - **leaf nodes** represent the input relations;
  - **internal nodes** represent the intermediate result;
  - **the root node** represents the resulting relation.
- **Example:**

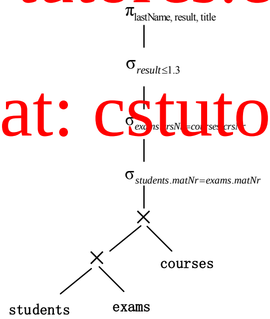
$$\pi_{lastName, result, title}(\sigma_{result \leq 1.3}((\texttt{Students} \bowtie_{\texttt{Students}.matNr=\texttt{Exams}.matNr} \texttt{Exams}) \bowtie_{\texttt{Exams}.crsNr=\texttt{Courses}.crsNr} \texttt{Courses}))$$



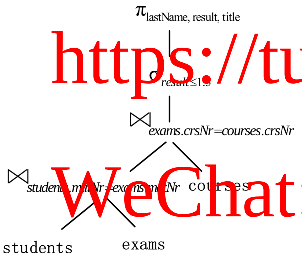
Query Tree Example

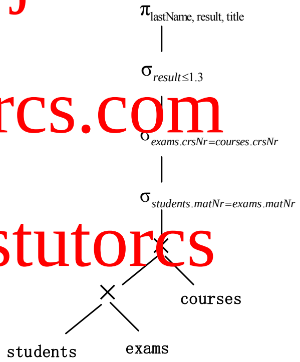
- For each query tree, computation proceeds **bottom-up**:
  - child nodes must be executed before their parent nodes;
  - but there can exist multiple methods of executing sibling nodes.
- **Example:**

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



Equivalent Query Trees (Query Optimisation)

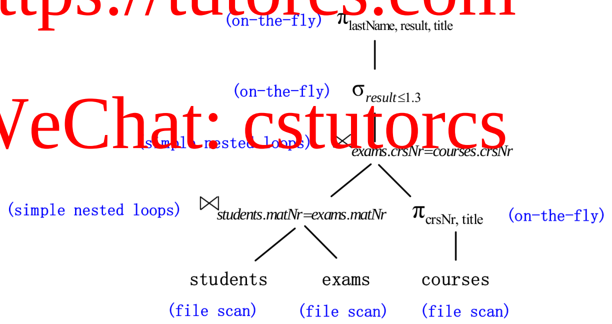




Execution Plan

(Slide 8-27 will not be assessed in our course)

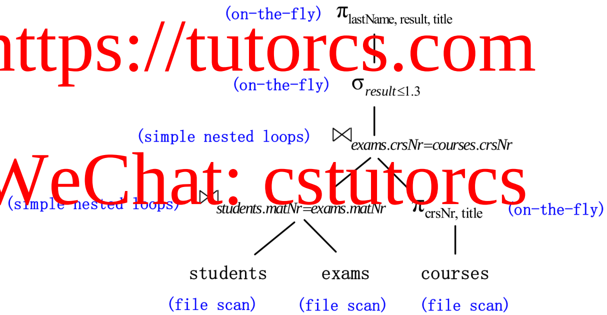
- A **query execution plan** consists of an (extended) query tree with additional annotation at each node indicating:
  - (1) the *access method* to use for each table, and
  - (2) the *implementation method* for each RA operator.



Australian National University

Execution Plan

- **Materialized**: The intermediate result of an operator may be saved in a temporary table for processing by the next operator.
- **Pipelined**: the intermediate result of an operator is directly sent to another operator without creating a temporary table (also called **on-the-fly**).



Note: Pipelined evaluation may have significant saving on I/O cost, while materialized evaluation can avoid repeated computations.

Execution Plan

- **Question:** Which execution plan is '**optimal**' in terms of processing efficiency?

Execution Plan

- **Question:** Which execution plan is '**optimal**' in terms of processing efficiency?

- This is determined by the query optimiser using a variety of algorithms (**Fact**: there is no true optimal solution in general!).

- Realistically, we cannot expect to always find the best plan, but we expect to **consistently find a plan that is good**.

Execution Plan

- **Question:** Which execution plan is '**optimal**' in terms of processing efficiency?
- This is determined by the query optimiser using a variety of algorithms (**Fact**: there is no true optimal solution in general!).

- Realistically, we cannot expect to always find the best plan, but we expect to **consistently find a plan that is good**.

- The **performance** of different execution plans for the same query may differ considerably (e.g., seconds vs. hours vs. days):
  - different but equivalent **RA expressions**;
  - different algorithms for **each RA operator**.

Execution Plan

- Basic ideas of algorithms used for RA operators

Execution Plan

- Basic ideas of algorithms used for RA operators
  - **Selection:** If there is no index, we have to scan the table. Otherwise, we scan the indexes to retrieve matching tuples and apply remaining selection conditions to further restrict the tuples.

Australian National University

Execution Plan

- Basic ideas of algorithms used for RA operators
  - **Selection:** If there is no index, we have to scan the table. Otherwise, we scan the indexes to retrieve matching tuples and apply remaining selection conditions to further restrict the tuples.
  - **Projection:** retrieves a subset of attributes from each tuple of the table (similar to selection). If requiring duplicate elimination, then we have to do sorting additionally (expensive part!)

Australian National University

Execution Plan

- Basic ideas of algorithms used for RA operators
  - **Selection:** If there is no index, we have to scan the table. Otherwise, we scan the indexes to retrieve matching tuples and apply remaining selection conditions to further restrict the tuples.
  - **Projection:** retrieves a subset of attributes from each tuple of the table (similar to selection). If requiring duplicate elimination, then we have to do sorting additionally (expensive part!)
  - **Join:** We may use *nested loops join*, or *sort-merge join*, *hash joins*, etc.

Execution Plan

- Basic ideas of algorithms used for RA operators
  - **Selection:** If there is no index, we have to scan the table. Otherwise, we scan the indexes to retrieve matching tuples and apply remaining selection conditions to further restrict the tuples.
  - **Projection:** retrieves a subset of attributes from each tuple of the table (similar to selection). If requiring duplicate elimination, then we have to do sorting additionally (expensive part!)
  - **Join:** We may use *nested loops join*, or *sort-merge join*, *hash joins*, etc.
  - **Group by** and **order by** are typically implemented using sorting.

Execution Plan

- Basic ideas of algorithms used for RA operators
  - **Selection:** If there is no index, we have to scan the table. Otherwise, we scan the indexes to retrieve matching tuples and apply remaining selection conditions to further restrict the tuples.
  - **Projection:** retrieves a subset of attributes from each tuple of the table (similar to selection). If requiring duplicate elimination, then we have to do sorting additionally (expensive part!)
  - **Join:** We may use *nested loops join*, or *sort-merge join*, *hash joins*, etc.
  - **Group by** and **order by** are typically implemented using sorting.
  - **Aggregation operators** use temporary counters in main memory when retrieving tuples.

Execution Plan

- Basic ideas of algorithms used for RA operators
  - **Selection:** If there is no index, we have to scan the table. Otherwise, we scan the indexes to retrieve matching tuples and apply remaining selection conditions to further restrict the tuples.
  - **Projection:** retrieves a subset of attributes from each tuple of the table (similar to selection). If requiring duplicate elimination, then we have to do sorting additionally (expensive part!)
  - **Join:** We may use *nested loops join*, or *sort-merge join*, *hash joins*, etc.
  - **Group by** and **order by** are typically implemented using sorting.
  - **Aggregation operators** use temporary counters in main memory when retrieving tuples.
  - **Set operators** can use the same approach as projection to eliminate duplicates.

Estimating Query Costs - Example

- Which movies got a non-US award for one of its actors playing an *'agent'*?

Estimating Query Costs - Example

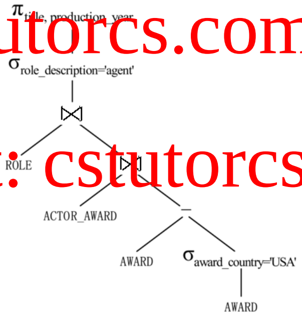
- Which movies got a non-US award for one of its actors playing an 'agent'?

$$\pi_{title,production\_year}(\sigma_{role\_description='agent'}(\text{ROLE} \bowtie \text{ACTOR\_AWARD} \bowtie (\text{AWARD} - \sigma_{award\_country='USA'}(\text{AWARD}))))$$

Estimating Query Costs - Example

- Which movies got a non-US award for one of its actors playing an *'agent'*?

$$\pi_{title,production\_year}(\sigma_{role\_description='agent'}(\textsc{Role} \bowtie \textsc{Actor\_Award} \bowtie (\textsc{Award} - \sigma_{award\_country='USA'}(\textsc{Award}))))$$



Size of Relations

- How to determine the **size of a relation** r over $R(A_1, \ldots, A_k)$?
  - Let n denote the average number of tuples in r, and ℓ_j the the average space (e.g., in bits) for attribute A_j.

	R			
	A_1	A_2	$\ldots$	A_k
1	$\ldots$	$\ldots$	$\ldots$	$\ldots$
$\ldots$	$\ldots$	$\ldots$	$\ldots$	$\ldots$
n	$\ldots$	$\ldots$	$\ldots$	$\ldots$
	ℓ_1	ℓ_2	$\ldots$	ℓ_k

Size of Relations

- How to determine the **size of a relation** r over $R(A_1, \ldots, A_k)$?
  - Let n denote the average number of tuples in r, and ℓ_j the the average space (e.g., in bits) for attribute A_j.

	R			
	A_1	A_2	$\ldots$	A_k
1	$\ldots$	$\ldots$	$\ldots$	$\ldots$
$\ldots$	$\ldots$	$\ldots$	$\ldots$	$\ldots$
n	$\ldots$	$\ldots$	$\ldots$	$\ldots$
	ℓ_1	ℓ_2	$\ldots$	ℓ_k

  - Then, $n \cdot \sum_{j=1}^{k} \ell_j$ is the size of the relation r.
- We use this formula to assign sizes to leaf nodes in the query tree.

Estimating Query Costs - Example (Relation Sizes)

- AWARD(Award_name:`varchar(30)`,Institution:`varchar(50)`,Award_country:`varchar(20)`)

Estimating Query Costs - Example (Relation Sizes)

- AWARD(Award_name:`varchar`(30),Institution:`varchar`(50),Award_country: `varchar`(20))
  - Estimate the average number of tuples as 15.

Estimating Query Costs - Example (Relation Sizes)

- AWARD(Award_name:`varchar`(30),Institution:`varchar`(50),Award_country: `varchar`(20))
  - Estimate the average number of tuples as 15.
  - Estimate the average space for attributes:

Estimating Query Costs - Example (Relation Sizes)

- AWARD(Award_name:`varchar`(30),Institution:`varchar`(50),Award_country: `varchar`(20))

- Estimate the average number of tuples as 15.
- Estimate the average space for attributes:
  - Award_name: $8 \cdot 20 = 160$ bits (the mean length is 20);
  - Institution: $8 \cdot 30 = 240$ bits (the mean length is 30);
  - Award_country: $8 \cdot 10 = 80$ bits (the mean length is 10).

Estimating Query Costs - Example (Relation Sizes)

- AWARD(Award_name:`varchar`(30),Institution:`varchar`(50),Award_country: `varchar`(20))
  - Estimate the average number of tuples as 15.

  - Estimate the average space for attributes:
    - Award_name: $8 \cdot 20 = 160$ bits (the mean length is 20);
    - Institution: $8 \cdot 30 = 240$ bits (the mean length is 30);

    - Award_country: $8 \cdot 10 = 80$ bits (the mean length is 10).
- The average size of a tuple is $160 + 80 + 240 = 480$ bits.

Estimating Query Costs - Example (Relation Sizes)

- AWARD(Award_name:`varchar`(30),Institution:`varchar`(50),Award_country: `varchar`(20))
  - Estimate the average number of tuples as 15.
  - Estimate the average space for attributes:
    - Award_name: $8 \cdot 20 = 160$ bits (the mean length is 20);
    - Institution: $8 \cdot 30 = 240$ bits (the mean length is 30);
    - Award_country: $8 \cdot 10 = 80$ bits (the mean length is 10).
- The average size of a tuple is $160 + 80 + 240 = 480$ bits.
- The average size of a relation is estimated to be $15 \cdot 480 = 7,200$ bits.

Estimating Query Costs - Example (Relation Sizes)

- ROLE(Id:`char`(8), Title:`varchar`(40), Production_year:`number`(4), Role_description:`varchar`(100),Credits:`varchar`(40))

Estimating Query Costs - Example (Relation Sizes)

- ROLE(Id:`char(8)`, Title:`varchar(40)`, Production_year:`number(4)`, Role_description:`varchar(100)`,Credits:`varchar(40)`)
  - Estimate the average number of tuples as 500.

Estimating Query Costs - Example (Relation Sizes)

- ROLE(Id:char(8), Title:varchar(40), Production_year:number(4), Role_description:varchar(100),Credits:varchar(40))
  - Estimate the average number of tuples as 500.
  - Estimate the average space for attributes:

Estimating Query Costs - Example (Relation Sizes)

- ROLE(Id:`char`(8), Title:`varchar`(40), Production_year:`number`(4), Role_description:`varchar`(100),Credits:`varchar`(40))
  - Estimate the average number of tuples as 500.
  - Estimate the average space for attributes:
    - Id: $8 \cdot 8 = 64$ bits (as the domain is char(8));
    - Title: $8 \cdot 25 = 200$ bits (the mean length is 25);
    - Production_year: 13 bits (as the domain is number(4));
    - Role_description: $8 \cdot 50 = 400$ bits (the mean length is 50);
    - Credits: $8 \cdot 20 = 160$ bits (the mean length is 20).

Estimating Query Costs - Example (Relation Sizes)

- ROLE(Id:`char`(8), Title:`varchar`(40), Production_year:`number`(4), Role_description:`varchar`(100),Credits:`varchar`(40))
  - Estimate the average number of tuples as 500.

  - Estimate the average space for attributes:
    - Id: $8 \cdot 8 = 64$ bits (as the domain is char(8));
    - Title: $8 \cdot 25 = 200$ bits (the mean length is 25);
    - Production_year: 13 bits (as the domain is number(4));

    - Role_description: $8 \cdot 50 = 400$ bits (the mean length is 50);
    - Credits: $8 \cdot 20 = 160$ bits (the mean length is 20).
- The average size of a tuple is $64 + 200 + 13 + 400 + 160 = 837$ bits

Estimating Query Costs - Example (Relation Sizes)

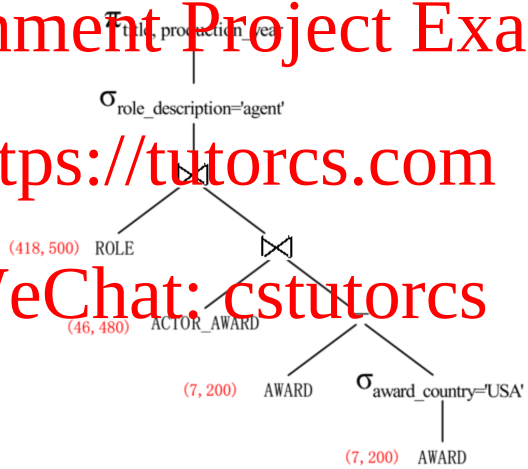
- ROLE(Id:`char`(8), Title:`varchar`(40), Production_year:`number`(4), Role_description:`varchar`(100),Credits:`varchar`(40))
  - Estimate the average number of tuples as 500.
  - Estimate the average space for attributes:
    - Id: $8 \cdot 8 = 64$ bits (as the domain is char(8));
    - Title: $8 \cdot 25 = 200$ bits (the mean length is 25);
    - Production_year: 13 bits (as the domain is number(4));
    - Role_description: $8 \cdot 50 = 400$ bits (the mean length is 50);
    - Credits: $8 \cdot 20 = 160$ bits (the mean length is 20).

- The average size of a tuple is $64 + 200 + 13 + 400 + 160 = 837$ bits
- The average size of a relation is to be $500 \cdot 837 = 418,500$ bits

Estimating Query Costs - Example (Relation Sizes)

- ACTOR_AWARD(Title:`varchar(40)`,Production_year:`number(4)`, Role_description:`varchar(100)`,Award_name:`varchar(30)`, Year_of_award:`number(4)`,Category:`varchar(100)`,Result:`varchar(20)`)
  - Estimate the average number of tuples as 40
  - Estimate the average space for attributes:
    - Title: 200 bits (as before);
    - Production_year: 13 bits (as before);
    - Role_description: 400 bits (as before);
    - Award_name: 160 bits (as before);
    - Year_of_award: 13 bits (as the domain is number(4));
    - Category: $8 \cdot 40 = 320$ bits (the mean length is 40);
    - Result: $8 \cdot 7 = 56$ bits (the mean length is 7).
- The average size of a tuple is $200 + 13 + 400 + 160 + 13 + 320 + 56 = 1,162$ bits.
- The average size of a relation is $40 \cdot 1162 = 46,480$ bits.

Estimating Query Costs - Example (Query Tree)



Size of Selection Node

- Selection σ_φ is linear in the number n of tuples of the involved relation:
  - Scan the relation one tuple after another (if there is no index);
  - Check for each tuple, whether the condition φ is satisfied or not;
  - Keep exactly those tuples satisfying φ.

Size of Selection Node

- Selection σ_φ is linear in the number n of tuples of the involved relation:
  - Scan the relation one tuple after another (if there is no index);
  - Check for each tuple, whether the condition φ is satisfied or not;
  - Keep exactly those tuples satisfying φ.

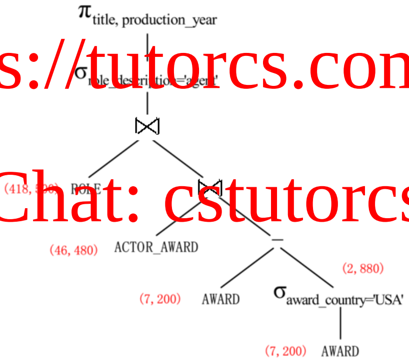
- Let s be the size of its single relevant node.
- The **size of a selection node** σ_φ is

$$\boxed{a_\varphi \cdot s},$$

where a_φ is the average percentage of tuples satisfying φ.

Estimating Query Costs - Example (Selection)

- For selection $\sigma_{award\_country=\text{'USA'}}$, assume $a_\varphi = 0.4$ (i.e., 40% of the movie awards from the USA). Hence, we have: $a_\varphi \cdot s = 0.4 \cdot 7{,}200 = 2{,}880$.



Size of Difference Node

- Let s_1 and s_2 be the sizes of the two relevant nodes.
- Again, we need to consider **the probability** that tuples occur in both relations.

Size of Difference Node

- Let s_1 and s_2 be the sizes of the two relevant nodes.

- Again, we need to consider **the probability** that tuples occur in both relations.

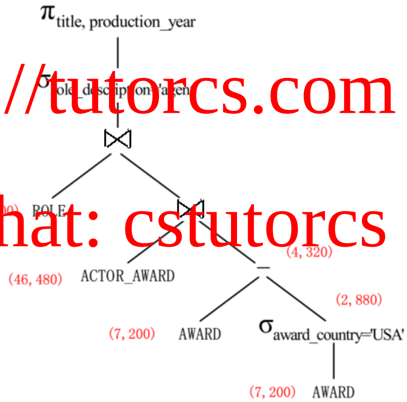
- The **size of a difference node** is

$$s_1 \cdot (1 - p)$$

where $(1 - p)$ is the probability that tuples from s_1 does not occur in s_2.

Estimating Query Costs - Example (Difference)

- Since 40% of the movie awards from the USA, the probability of an award to be a US-award is $p = 0.4$. We have: $s_1 \cdot (1 - p) = 7{,}200 \cdot (1\text{-}0.4) = 4{,}320$.



Size of Natural Join Node

- Let s_1 and s_2 be the sizes of the two relevant nodes, and r_1 and r_2 be the size of a tuple in these two nodes. $\frac{s_1}{r_1}$ and $\frac{s_2}{r_2}$ are the estimated number of tuples in these two nodes.

Size of Natural Join Node

- Let s_1 and s_2 be the sizes of the two relevant nodes, and r_1 and r_2 be the size of a tuple in these two nodes. $\frac{s_1}{r_1}$ and $\frac{s_2}{r_2}$ are the estimated number of tuples in these two nodes.

- The **size of a natural join node** is

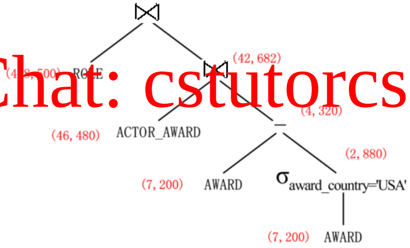
$$\frac{s_1}{r_1} \cdot p \cdot \frac{s_2}{r_2} \cdot (r_1 + r_2 - r),$$

where r is the size of a tuple over the **common attributes**, and p is the **matching probability** (for any tuple of the first relevant node to match with any tuples in the second relevant relation). Note that $r_1 + r_2 - r$ is the size of a tuple after the natural join operation.

Estimating Query Costs - Example (Natural Join)

- For join with ACTOR_AWARD assume $p = 0.08$, i.e., 8% of the actor awards are non-US awards. By $\frac{s_1}{r_1} \cdot p \cdot \frac{s_2}{r_2} \cdot (r_1 + r_2 - r)$, we have:

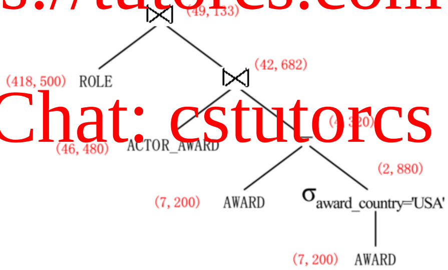
$$\frac{46,480}{1,162} \cdot 0.08 \cdot \frac{4,320}{480} \cdot (1,162 + 480 - 160) = 42,682.$$



Estimating Query Costs - Example (Natural Join)

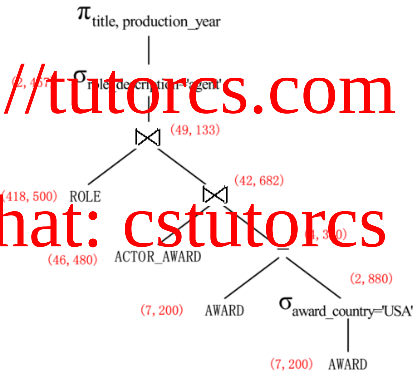
- Assume $p = 0.002$. By $\frac{s_1}{r_1} \cdot p \cdot \frac{s_2}{r_2} \cdot (r_1 + r_2 - r)$, we have:

$$\frac{418,500}{837} \cdot 0.002 \cdot \frac{42,682}{1,482} \cdot (837 + 1,482 - 200 - 400 - 13) = 49,133.$$



Estimating Query Costs - Example (Selection)

- For selection $\sigma_{role\_description='agent'}$, assume $a_\varphi = 0.05$ (i.e., non-US awards for "agent" roles are 5%). Hence, we have: $a_\varphi \cdot s = 0.05 \cdot 49{,}133 = 2{,}457$.



Size of Projection Node

- Projection $\pi_{\{A_1,\ldots,A_n\}}$:
  - Project each tuple to the attributes in $\{A_1, \ldots, A_n\}$
  - Eliminate duplicates (**Note**: SQL does not eliminate tuples unless `DISTINCT` is used).

Size of Projection Node

- Projection $\pi_{\{A_1,\dots,A_n\}}$:
  - Project each tuple to the attributes in $\{A_1, \dots, A_n\}$
  - Eliminate duplicates (**Note**: SQL does not eliminate tuples unless `DISTINCT` is used).

- Let s be the size of its single relevant node with $s = n \cdot r$ for its average number n of tuples and its average size r of a tuple.

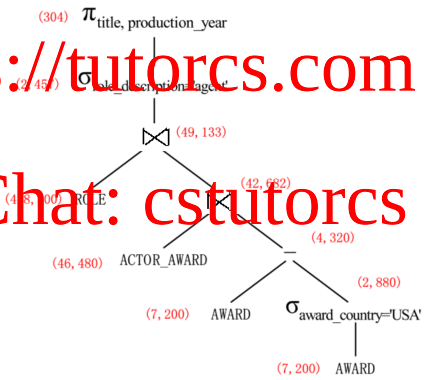
- The **size of a projection node** $\pi_{A_1,\dots,A_n}$ is

$$(1 - p_i) \cdot s \cdot \frac{r_i}{r},$$

where r_i is the average size of a tuple over $\{A_1, \dots, A_n\}$, and p_i is the probability that two tuples coincide on $A_1, \dots, A_n$ (i.e., the same values on all attributes $A_1, \dots, A_n$).

Estimating Query Costs - Example (Projection)

- For projection $\pi_{title, production\_year}$, assume that there are 1% of duplicates, i.e., $p_i = 0.01$. By $(1 - p_i) \cdot s \cdot \frac{r_i}{r}$, we have $(1 - 0.01) \cdot 2,457 \frac{213}{1706} = 304$

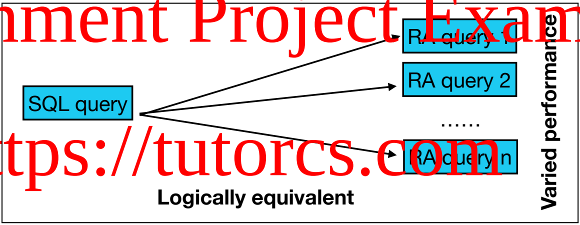


Query Optimisation

Are they still equivalent?

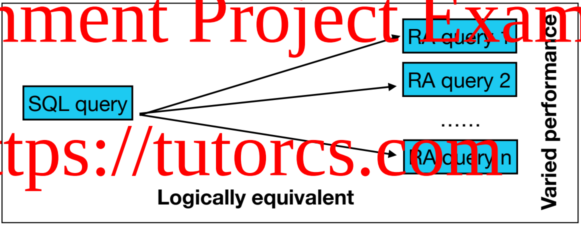
$$
\begin{aligned}
a &= b \\
a^2 &= ab \\
a^2 - b^2 &= ab - b^2 \\
(a+b)(a-b) &= b(a-b) \\
a+b &= b \\
2b &= b \\
2 &= 1
\end{aligned}
$$

Relational Algebra ⇒ Query Optimisation



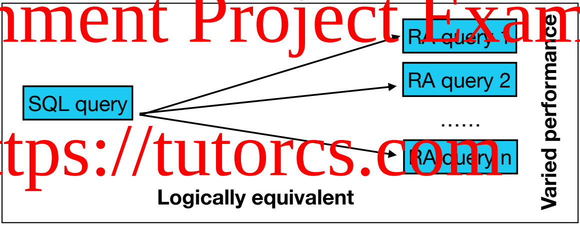
- Which RA query should be chosen for a given SQL query?

Relational Algebra ⇒ Query Optimisation



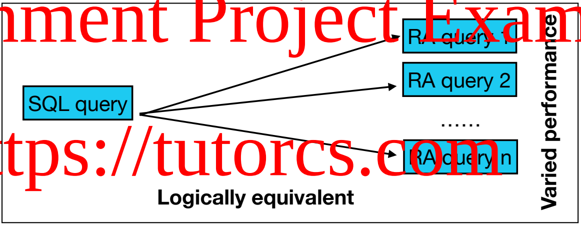
- Which RA query should be chosen for a given SQL query?
  - **Who choose?**

Relational Algebra ⇒ Query Optimisation



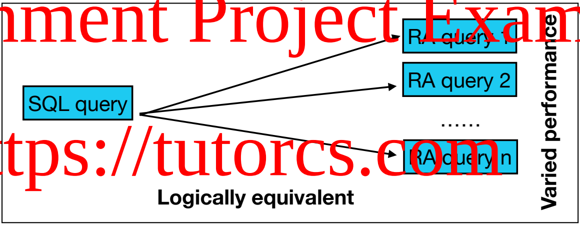
- Which RA query should be chosen for a given SQL query?
  - **Who choose?** Query optimiser!

Relational Algebra ⇒ Query Optimisation



- Which RA query should be chosen for a given SQL query?
  - **Who choose?** Query optimiser!
  - **How to choose?**

Relational Algebra ⇒ Query Optimisation



- Which RA query should be chosen for a given SQL query?
  - **Who choose?** Query optimiser!
  - **How to choose?**
    - Semantic query optimisation
    - Rule-based optimisation
    - Cost-based optimisation

Query Optimisation

- In practice, query optimisers incorporate elements of the following three optimisation approaches:

Query Optimisation

- In practice, query optimisers incorporate elements of the following three optimisation approaches:
  - **Semantic query optimisation**

    Use application specific semantic knowledge to transform a query into the one with a lower cost (they return the same answer).

Australian National University

Query Optimisation

- In practice, query optimisers incorporate elements of the following three optimisation approaches:
  - **Semantic query optimisation**

    Use application specific semantic knowledge to transform a query into the one with a lower cost (they return the same answer).
  - **Rule-based query optimisation**

    Use heuristic rules to transform a relational algebra expression into an equivalent one with a possibly lower cost.

Query Optimisation

- In practice, query optimisers incorporate elements of the following three optimisation approaches:
  - **Semantic query optimisation**

    Use application specific semantic knowledge to transform a query into the one with a lower cost (they return the same answer).
  - **Rule-based query optimisation**

    Use heuristic rules to transform a relational algebra expression into an equivalent one with a possibly lower cost.

  - **Cost-based query optimisation**

    Use a cost model to estimate the costs of plans, and then select the most cost-effective plan. This will not be assessed in our course.

Semantic Query Optimisation

- **Example**:

  PERSON(id, first_name, last_name, year_born)
  MOVIE(title, production_year, country, run_time, major_genre)
  WRITER(id, title, production_year, credits) where

  [id] $\subseteq$ PERSON[id]
  [title, production_year] $\subseteq$ MOVIE [title, production_year]

- List the ids of the writers who have written movies produced in 2000.

Semantic Query Optimisation

- **Example**:

  PERSON(id, first_name, last_name, year_born)
  MOVIE(title, production_year, country, run_time, major_genre)
  WRITER(id, title, production_year, credits) where

  [id] ⊆ PERSON[id]
  [title, production_year] ⊆ MOVIE [title, production_year]

- List the ids of the writers who have written movies produced in 2000.

$\pi_{id}\sigma_{production\_year=2000}(\text{WRITER} \bowtie \text{PERSON} \bowtie \text{MOVIE})$

$\pi_{id}\sigma_{production\_year=2000}(\text{WRITER} \bowtie \text{PERSON})$

$\pi_{id}\sigma_{production\_year=2000}(\text{WRITER} \bowtie \text{MOVIE})$

Australian National University

Semantic Query Optimisation

- **Example**:

  PERSON(id, first_name, last_name, year_born)
  MOVIE(title, production_year, country, run_time, major_genre)
  WRITER(id, title, production_year, credits) where

  [id] ⊆ PERSON[id]
  [title, production_year] ⊆ MOVIE [title, production_year]

- List the ids of the writers who have written movies produced in 2000.

$\pi_{id}\sigma_{production\_year=2000}(\text{WRITER} \bowtie \text{PERSON} \bowtie \text{MOVIE})$

$\pi_{id}\sigma_{production\_year=2000}(\text{WRITER} \bowtie \text{PERSON})$

$\pi_{id}\sigma_{production\_year=2000}(\text{WRITER} \bowtie \text{MOVIE})$

- $\pi_{id}\sigma_{production\_year=2000}\text{WRITER}$ $\longleftarrow$ the optimised RA

Rule-based Query Optimisation

- A rule-based optimisation transforms the RA expression by using a set of heuristic rules that typically improve the execution performance.

Rule-based Query Optimisation

- A rule-based optimisation transforms the RA expression by using a set of heuristic rules that typically improve the execution performance.

- **Key ideas:** apply the most restrictive operation before other operations, which can reduce the size of intermediate results.

Rule-based Query Optimisation

- A rule-based optimisation transforms the RA expression by using a set of heuristic rules that typically improve the execution performance.

- **Key ideas:** apply the most restrictive operation before other operations, which can reduce the size of intermediate results.
  - **Push-down selection:**

    Apply as early as possible to reduce the number of tuples;

Rule-based Query Optimisation

- A rule-based optimisation transforms the RA expression by using a set of heuristic rules that typically improve the execution performance.

- **Key ideas:** apply the most restrictive operation before other operations, which can reduce the size of intermediate results.
  - **Push-down selection:**

    Apply as early as possible to reduce the number of tuples;

  - **Push-down projection:**

    Apply as early as possible to reduce the number of attributes.

Rule-based Query Optimisation

- A rule-based optimisation transforms the RA expression by using a set of heuristic rules that typically improve the execution performance.

- **Key ideas:** apply the most restrictive operation before other operations, which can reduce the size of intermediate results.
  - **Push-down selection:**

    Apply as early as possible to reduce the number of tuples;

  - **Push-down projection:**

    Apply as early as possible to reduce the number of attributes.

Rule-based Query Optimisation

- A rule-based optimisation transforms the RA expression by using a set of heuristic rules that typically improve the execution performance.
- **Key ideas:** apply the most restrictive operation before other operations, which can reduce the size of intermediate results.

  - **Push-down selection:**

    Apply as early as possible to reduce the number of tuples;

  - **Push-down projection:**

    Apply as early as possible to reduce the number of attributes.

- But we must ensure that the resulting query tree gives the same result as the original query tree, i.e., **the equivalence of RA expressions**.

Australian National University

Rule-based Optimisation

- Can they be executed in one go? $\hookrightarrow$ Merging RA operators.

Australian National University

Rule-based Optimisation

- Can they be executed in one go? $\hookrightarrow$ Merging RA operators.
  - $\sigma_{\varphi}(\sigma_{\psi}(R)) \equiv \sigma_{\varphi \wedge \psi}(R)$;
  - $\pi_X(\pi_Y(R)) \equiv \pi_X(R)$ if $X \subseteq Y$;
  - $\sigma_{\varphi}(R_1 \times R_2) \equiv R_1 \bowtie_{\varphi} R_2$;
  - $\sigma_{\varphi_1}(R_1 \bowtie_{\varphi_2} R_2) \equiv R_2 \bowtie_{\varphi_1 \wedge \varphi_2} R_1$;

Rule-based Optimisation

- Can they be executed in one go? $\hookrightarrow$ Merging RA operators

Rule-based Optimisation

- Can they be executed in one go? $\hookrightarrow$ Merging RA operators
  - $\sigma_{\varphi}(\sigma_{\psi}(R)) \equiv \sigma_{\varphi \wedge \psi}(R)$;

Rule-based Optimisation

- Can they be executed in one go? $\hookrightarrow$ Merging RA operators.
  - $\sigma_{\varphi}(\sigma_{\psi}(R)) \equiv \sigma_{\varphi \wedge \psi}(R)$;

$\sigma_{CourseNo=`COMP2400'}(\sigma_{UID=111}(STUDY))$ v.s. $\sigma_{(Course=`COMP2400')\wedge(UID=111)}(STUDY)$

Rule-based Optimisation

- Can they be executed in one go? $\hookrightarrow$ Merging RA operators
  - $\sigma_{\varphi}(\sigma_{\psi}(R)) \equiv \sigma_{\varphi \wedge \psi}(R)$;

$\sigma_{CourseNo=\text{'}COMP2400\text{'}}(\sigma_{UID=111}(STUDY))$ v.s. $\sigma_{(Course=\text{'}COMP2400\text{'})\wedge(UID=111)}(STUDY)$

STUDY		
UID	CourseNo	Hours
111	COMP2400	120
222	COMP2400	115
333	STAT2001	120
111	BUSN2011	110
111	ECON2102	120
333	BUSN2011	130

STUDY		
UID	CourseNo	Hours
111	COMP2400	120
111	BUSN2011	110
111	ECON2102	120

STUDY		
UID	CourseNo	Hours
111	COMP2400	120

Rule-based Optimisation

- Can they be executed in one go? $\hookrightarrow$ Merging RA operators

- $\sigma_{\varphi}(\sigma_{\psi}(R)) \equiv \sigma_{\varphi \wedge \psi}(R)$;

$\sigma_{CourseNo='COMP2400'}(\sigma_{UID=111}(STUDY))$ v.s. $\sigma_{(Course='COMP2400')\wedge(UID=111)}(STUDY)$

STUDY		
UID	CourseNo	Hours
111	COMP2400	120
222	COMP2400	115
333	STAT2001	120
111	BUSN2011	110
111	ECON2102	120
333	BUSN2011	130

STUDY		
UID	CourseNo	Hours
111	COMP2400	120
111	BUSN2011	110
111	ECON2102	120

STUDY		
UID	CourseNo	Hours
111	COMP2400	120

STUDY		
UID	CourseNo	Hours
111	COMP2400	120
222	COMP2400	115
333	STAT2001	120
111	BUSN2011	110
111	ECON2102	120
333	BUSN2011	130

$\Rightarrow$

(without any intermediate relation)

STUDY		
UID	CourseNo	Hours
111	COMP2400	120

Rule-based Optimisation

- Can they be executed in one go? $\hookrightarrow$ Merging RA operators

Rule-based Optimisation

- Can they be executed in one go? $\hookrightarrow$ Merging RA operators
  - $\pi_X(\pi_Y(R)) \equiv \pi_X(R)$ if $X \subseteq Y$;

Rule-based Optimisation

- Can they be executed in one go? $\hookrightarrow$ Merging RA operators
  - $\pi_X(\pi_Y(R)) \equiv \pi_X(R)$ if $X \subseteq Y$;

$\pi_{UID}(\pi_{UID,CourseNo}(Study))$ v.s. $\pi_{UID}(Study)$

Rule-based Optimisation

- Can they be executed in one go? $\hookrightarrow$ Merging RA operators
  - $\pi_X(\pi_Y(R)) \equiv \pi_X(R)$ if $X \subseteq Y$;

$\pi_{UID}(\pi_{UID,CourseNo}(Study))$ v.s. $\pi_{UID}(Study)$

Study

UID	CourseNo	Hours
111	COMP2400	120
222	COMP2400	115
333	STAT2001	120
111	BUSN2011	110
111	ECON2102	120
333	BUSN2011	130

UID	CourseNo
111	COMP2400
222	COMP2400
333	STAT2001
111	BUSN2011
111	ECON2102
333	BUSN2011

UID
111
222
333

Rule-based Optimisation

- Can they be executed in one go? $\hookrightarrow$ Merging RA operators
  - $\pi_X(\pi_Y(R)) \equiv \pi_X(R)$ if $X \subseteq Y$;

$\pi_{UID}(\pi_{UID,CourseNo}(Study))$ v.s. $\pi_{UID}(Study)$

STUDY		
UID	CourseNo	Hours
111	COMP2400	120
222	COMP2400	115
333	STAT2001	120
111	BUSN2011	110
111	ECON2102	120
333	BUSN2011	130

UID	CourseNo
111	COMP2400
222	COMP2400
333	STAT2001
111	BUSN2011
111	ECON2102
333	BUSN2011

UID
111
222
333

STUDY		
UID	CourseNo	Hours
111	COMP2400	120
222	COMP2400	115
333	STAT2001	120
111	BUSN2011	110
111	ECON2102	120
333	BUSN2011	130

$\Rightarrow$

(without any intermediate relation)

UID
111
222
333

Rule-based Optimisation

- Can they be executed in one go? $\rightsquigarrow$ Merging RA operators.

Australian National University

Rule-based Optimisation

- Can they be executed in one go? $\rightsquigarrow$ Merging RA operators.
  - $\sigma_{\varphi}(R_1 \times R_2) \equiv R_1 \bowtie_{\varphi} R_2$

Rule-based Optimisation

- Can they be executed in one go? $\rightsquigarrow$ Merging RA operators.
  - $\sigma_{\varphi}(R_1 \times R_2) \equiv R_1 \bowtie_{\varphi} R_2$

    $\sigma_{Course.No=Enrol.CoureNo}(Course \times Enrol)$

Rule-based Optimisation

- Can they be executed in one go? $\rightsquigarrow$ Merging RA operators.
  - $\sigma_\varphi(R_1 \times R_2) \equiv R_1 \bowtie_\varphi R_2$

    $\sigma_{Course.No=Enrol.CoureNo}(Course \times Enrol)$

COURSE

No	Cname	Unit
COMP2400	Relational Databases	6
BUSN2011	Management Accounting	6

ENROL

StudentID	CourseNo	Semester	Status
111	BUSN2011	2016 S1	active
222	COMP2400	2016 S1	active
111	COMP2400	2016 S2	active

Rule-based Optimisation

- Can they be executed in one go? $\leadsto$ Merging RA operators.
  - $\sigma_{\varphi}(R_1 \times R_2) \equiv R_1 \bowtie_{\varphi} R_2$

    $\sigma_{Course.No=Enrol.CoureNo}(Course \times Enrol)$

COURSE

No	Cname	Unit
COMP2400	Relational Databases	6
BUSN2011	Management Accounting	6

ENROL

StudentID	CourseNo	Semester	Status
111	BUSN2011	2016 S1	active
222	COMP2400	2016 S1	active
111	COMP2400	2016 S2	active

No	Cname	Unit	StudentID	CourseNo	Semester	Status
COMP2400	Relational Databases	6	111	BUSN2011	2016 S1	active
COMP2400	Relational Databases	6	222	COMP2400	2016 S1	active
COMP2400	Relational Databases	6	111	COMP2400	2016 S2	active
BUSN2011	Management Accounting	6	111	BUSN2011	2016 S1	active
BUSN2011	Management Accounting	6	222	COMP2400	2016 S1	active
BUSN2011	Management Accounting	6	111	COMP2400	2016 S2	active

Rule-based Optimisation

- Can they be executed in one go? $\rightsquigarrow$ Merging RA operators.
  - $\sigma_{\varphi}(R_1 \times R_2) \equiv R_1 \bowtie_{\varphi} R_2$

$\sigma_{Course.No=Enrol.CoureNo}(Course \times Enrol)$

COURSE		
No	Cname	Unit
COMP2400	Relational Databases	6
BUSN2011	Management Accounting	6

ENROL			
StudentID	CourseNo	Semester	Status
111	BUSN2011	2016 S1	active
222	COMP2400	2016 S1	active
111	COMP2400	2016 S2	active

No	Cname	Unit	StudentID	CourseNo	Semester	Status
COMP2400	Relational Databases	6	111	BUSN2011	2016 S1	active
COMP2400	Relational Databases	6	222	COMP2400	2016 S1	active
COMP2400	Relational Databases	6	111	COMP2400	2016 S2	active
BUSN2011	Management Accounting	6	111	BUSN2011	2016 S1	active
BUSN2011	Management Accounting	6	222	COMP2400	2016 S1	active
BUSN2011	Management Accounting	6	111	COMP2400	2016 S2	active

Rule-based Optimisation

- Can they be executed in one go? $\rightsquigarrow$ Merging RA operators.
  - $\sigma_{\varphi}(R_1 \times R_2) \equiv R_1 \bowtie_{\varphi} R_2$

    $Course \bowtie_{Course.No=Enrol.CourseNo} Enrol$

COURSE		
No	Cname	Unit
COMP2400	Relational Databases	6
BUSN2011	Management Accounting	6

ENROL			
StudentID	CourseNo	Semester	Status
111	BUSN2011	2016 S1	active
222	COMP2400	2016 S1	active
111	COMP2400	2016 S2	active

Rule-based Optimisation

- Can they be executed in one go? $\rightsquigarrow$ Merging RA operators.
  - $\sigma_{\varphi}(R_1 \times R_2) \equiv R_1 \bowtie_{\varphi} R_2$

    $Course \bowtie_{Course.No=Enrol.CourseNo} Enrol$

COURSE		
No	Cname	Unit
COMP2400	Relational Databases	6
BUSN2011	Management Accounting	6

ENROL			
StudentID	CourseNo	Semester	Status
111	BUSN2011	2016 S1	active
222	COMP2400	2016 S1	active
111	COMP2400	2016 S2	active

Inner Join on Course.No=Enrol.CourseNo (no intermediate Cartesian product)

No	Cname	Unit	StudentID	CourseNo	Semester	Status
COMP2400	Relational Databases	6	222	COMP2400	2016 S1	active
COMP2400	Relational Databases	6	111	COMP2400	2016 S2	active
BUSN2011	Management Accounting	6	111	BUSN2011	2016 S1	active

Rule-based Optimisation

- Can join be executed last? $\hookrightarrow$ Push select/project before join.

Rule-based Optimisation

- Can join be executed last? $\hookrightarrow$ Push select/project before join.
  - $\sigma_{\varphi}(R_1 \bowtie R_2) \equiv \sigma_{\varphi}(R_1) \bowtie R_2$, if φ contains only attributes in R_1;
  - $\sigma_{\varphi_1 \wedge \varphi_2}(R_1 \bowtie R_2) \equiv \sigma_{\varphi_1}(R_1) \bowtie \sigma_{\varphi_2}(R_2)$, if φ_1 contains only attributes in R_1 and φ_2 contains only attributes in R_2;
  - $\pi_X(R_1 \bowtie R_2) \equiv \pi_X(\pi_{X_1}(R_1) \bowtie \pi_{X_2}(R_2))$, if the join condition contains attributes not in X, where X_i contains attributes both in R_i and X, and ones both in R_1 and R_2;
  - $\pi_X(R_1 \bowtie R_2) \equiv \pi_{X_1}(R_1) \bowtie \pi_{X_2}(R_2)$, if the join condition involves only attributes in X, where X_i contains attributes both in R_i and X, and ones both in R_1 and R_2;

Rule-based Optimisation

- Can join be executed last? $\hookrightarrow$ Push select/project before join
  - $\sigma_{\varphi}(R_1 \bowtie R_2) \equiv \sigma_{\varphi}(R_1) \bowtie R_2$, if φ contains only attributes in R_1;

$\sigma_{Cname=\text{'}ManagementAccounting'}(Course \bowtie Enrol)$

COURSE		
CourseNo	Cname	Unit
COMP2400	Relational Databases	6
BUSN2011	Management Accounting	6

Rule-based Optimisation

- Can join be executed last? $\hookrightarrow$ Push select/project before join
  - $\sigma_\varphi(R_1 \bowtie R_2) \equiv \sigma_\varphi(R_1) \bowtie R_2$, if φ contains only attributes in R_1;

$\sigma_{Cname=\text{'}ManagementAccounting\text{'}}(Course \bowtie Enrol)$

COURSE		
CourseNo	Cname	Unit
COMP2400	Relational Databases	6
BUSN2011	Management Accounting	6

ENROL			
StudentID	CourseNo	Semester	Status
111	BUSN2011	2016 S1	active
222	COMP2400	2016 S1	active
111	COMP2400	2016 S2	active

Rule-based Optimisation

- Can join be executed last? $\rightsquigarrow$ Push select/project before join
  - $\sigma_{\varphi}(R_1 \bowtie R_2) \equiv \sigma_{\varphi}(R_1) \bowtie R_2$, if φ contains only attributes in R_1;

$\sigma_{Cname=\text{'}ManagementAccounting\text{'}}(Course \bowtie Enrol)$

COURSE

CourseNo	Cname	Unit
COMP2400	Relational Databases	6
BUSN2011	Management Accounting	6

ENROL

StudentID	CourseNo	Semester	Status
111	BUSN2011	2016 S1	active
222	COMP2400	2016 S1	active
111	COMP2400	2016 S2	active

CourseNo	Cname	Unit	StudentID	Semester	Status
COMP2400	Relational Databases	6	222	2016 S1	active
COMP2400	Relational Databases	6	111	2016 S2	active
BUSN2011	Management Accounting	6	111	2016 S1	active

Australian National University

Rule-based Optimisation

- Can join be executed last? $\leadsto$ Push select/project before join
  - $\sigma_{\varphi}(R_1 \bowtie R_2) \equiv \sigma_{\varphi}(R_1) \bowtie R_2$, if φ contains only attributes in R_1;

$\sigma_{Cname=\text{'}ManagementAccounting\text{'}}(Course \bowtie Enrol)$

COURSE		
CourseNo	Cname	Unit
COMP2400	Relational Databases	6
BUSN2011	Management Accounting	6

ENROL			
StudentID	CourseNo	Semester	Status
111	BUSN2011	2016 S1	active
222	COMP2400	2016 S1	active
111	COMP2400	2016 S2	active

CourseNo	Cname	Unit	StudentID	Semester	Status
COMP2400	Relational Databases	6	222	2016 S1	active
COMP2400	Relational Databases	6	111	2016 S2	active
BUSN2011	Management Accounting	6	111	2016 S1	active

CourseNoNo	Cname	Unit	StudentID	Semester	Status
BUSN2011	Management Accounting	6	111	2016 S1	active

Australian National University

Rule-based Optimisation

- Can join be executed last? $\hookrightarrow$ Push select/project before join
  - $\sigma_{\varphi}(R_1 \bowtie R_2) \equiv \sigma_{\varphi}(R_1) \bowtie R_2$, if φ contains only attributes in R_1;

    $\sigma_{Cname=\text{'}ManagementAccounting\text{'}}(Course) \bowtie Enrol$

Rule-based Optimisation

- Can join be executed last? $\hookrightarrow$ Push select/project before join
  - $\sigma_{\varphi}(R_1 \bowtie R_2) \equiv \sigma_{\varphi}(R_1) \bowtie R_2$, if φ contains only attributes in R_1;

    $\sigma_{Cname='ManagementAccounting'}(Course) \bowtie Enrol$

COURSE		
CourseNo	Cname	Unit
COMP2400	Relational Databases	6
BUSN2011	Management Accounting	6

Rule-based Optimisation

- Can join be executed last? $\hookrightarrow$ Push select/project before join
  - $\sigma_{\varphi}(R_1 \bowtie R_2) \equiv \sigma_{\varphi}(R_1) \bowtie R_2$, if φ contains only attributes in R_1;

$\sigma_{Cname=\text{'ManagementAccounting'}}(Course) \bowtie Enrol$

COURSE		
CourseNo	Cname	Unit
COMP2400	Relational Databases	6
BUSN2011	Management Accounting	6

COURSE		
CourseNo	Cname	Unit
BUSN2011	ManagementAccounting	6

Rule-based Optimisation

- Can join be executed last? $\hookrightarrow$ Push select/project before join
  - $\sigma_{\varphi}(R_1 \bowtie R_2) \equiv \sigma_{\varphi}(R_1) \bowtie R_2$, if φ contains only attributes in R_1;

$\sigma_{Cname=\text{'}ManagementAccounting\text{'}}(Course) \bowtie Enrol$

COURSE		
CourseNo	Cname	Unit
COMP2400	Relational Databases	6
BUSN2011	Management Accounting	6

ENROL			
StudentID	CourseNo	Semester	Status
111	BUSN2011	2016 S1	active
222	COMP2400	2016 S1	active
111	COMP2400	2016 S2	active

COURSE		
CourseNo	Cname	Unit
BUSN2011	ManagementAccounting	6

Rule-based Optimisation

- Can join be executed last? $\hookrightarrow$ Push select/project before join
  - $\sigma_{\varphi}(R_1 \bowtie R_2) \equiv \sigma_{\varphi}(R_1) \bowtie R_2$, if φ contains only attributes in R_1;

$\sigma_{Cname=\text{'}ManagementAccounting\text{'}}(Course) \bowtie Enrol$

COURSE		
CourseNo	Cname	Unit
COMP2400	Relational Databases	6
BUSN2011	Management Accounting	6

ENROL			
StudentID	CourseNo	Semester	Status
111	BUSN2011	2016 S1	active
222	COMP2400	2016 S1	active
111	COMP2400	2016 S2	active

COURSE		
CourseNo	Cname	Unit
BUSN2011	Management Accounting	6

CourseNo	Cname	Unit	StudentID	Semester	Status
BUSN2011	Management Accounting	6	111	2016 S1	active

Rule-based Optimisation

- Can join be executed last? $\hookrightarrow$ Push select/project before join.

Rule-based Optimisation

- Can join be executed last? $\rightsquigarrow$ Push select/project before join.

  - $\pi_X(R_1 \bowtie R_2) \equiv \pi_{X_1}(R_1) \bowtie \pi_{X_2}(R_2)$, if the join condition involves only attributes in X, how could we derive X_1 and X_2?

Rule-based Optimisation

- Can join be executed last? $\leadsto$ Push select/project before join.

- $\pi_X(R_1 \bowtie R_2) \equiv \pi_{X_1}(R_1) \bowtie \pi_{X_2}(R_2)$, if the join condition involves only attributes in X, how could we derive X_1 and X_2?

  $\pi_{CourseNo,Cname,StudentID}(Course \bowtie Enrol)$

COURSE		
CourseNo	Cname	Unit
COMP2400	Relational Databases	6
BUSN2011	Management Accounting	6

Rule-based Optimisation

- Can join be executed last? $\leadsto$ Push select/project before join.
  - $\pi_X(R_1 \bowtie R_2) \equiv \pi_{X_1}(R_1) \bowtie \pi_{X_2}(R_2)$, if the join condition involves only attributes in X, how could we derive X_1 and X_2?

$\pi_{CourseNo,Cname,StudentID}(Course \bowtie Enrol)$

COURSE

CourseNo	Cname	Unit
COMP2400	Relational Databases	6
BUSN2011	Management Accounting	6

ENROL

StudentID	CourseNo	Semester	Status
111	BUSN2011	2016 S1	active
222	COMP2400	2016 S1	active
111	COMP2400	2016 S2	active

Rule-based Optimisation

- Can join be executed last? $\rightsquigarrow$ Push select/project before join.

  - $\pi_X(R_1 \bowtie R_2) \equiv \pi_{X_1}(R_1) \bowtie \pi_{X_2}(R_2)$, if the join condition involves only attributes in X, how could we derive X_1 and X_2?

    $\pi_{CourseNo,Cname,StudentID}(Course \bowtie Enrol)$

COURSE

CourseNo	Cname	Unit
COMP2400	Relational Databases	6
BUSN2011	Management Accounting	6

ENROL

StudentID	CourseNo	Semester	Status
111	BUSN2011	2016 S1	active
222	COMP2400	2016 S1	active
111	COMP2400	2016 S2	active

CourseNo	Cname	Unit	StudentID	Semester	Status
COMP2400	Relational Databases	6	222	2016 S1	active
COMP2400	Relational Databases	6	111	2016 S2	active
BUSN2011	Management Accounting	6	111	2016 S1	active

Rule-based Optimisation

- Can join be executed last? $\leadsto$ Push select/project before join.

- $\pi_X(R_1 \bowtie R_2) \equiv \pi_{X_1}(R_1) \bowtie \pi_{X_2}(R_2)$, if the join condition involves only attributes in X, how could we derive X_1 and X_2?

$\pi_{CourseNo,Cname,StudentID}(Course \bowtie Enrol)$

COURSE		
CourseNo	Cname	Unit
COMP2400	Relational Databases	6
BUSN2011	Management Accounting	6

ENROL			
StudentID	CourseNo	Semester	Status
111	BUSN2011	2016 S1	active
222	COMP2400	2016 S1	active
111	COMP2400	2016 S2	active

CourseNo	Cname	Unit	StudentID	Semester	Status
COMP2400	Relational Databases	6	222	2016 S1	active
COMP2400	Relational Databases	6	111	2016 S2	active
BUSN2011	Management Accounting	6	111	2016 S1	active

CourseNo	Cname	StudentID
COMP2400	Relational Databases	222
COMP2400	Relational Databases	111
BUSN2011	Management Accounting	111

Rule-based Optimisation

- Can join be executed last? $\leadsto$ Push select/project before join.

Rule-based Optimisation

- Can join be executed last? $\leadsto$ Push select/project before join.

  - $\pi_X(R_1 \bowtie R_2) \equiv \pi_{X_1}(R_1) \bowtie \pi_{X_2}(R_2)$, if the join condition involves only attributes in X, how could we derive X_1 and X_2?

Rule-based Optimisation

- Can join be executed last? $\rightsquigarrow$ Push select/project before join.

  - $\pi_X(R_1 \bowtie R_2) \equiv \pi_{X_1}(R_1) \bowtie \pi_{X_2}(R_2)$, if the join condition involves only attributes in X, how could we derive X_1 and X_2?

    $\pi_{CourseNo,Cname}(Course) \bowtie \pi_{CourseNo,StudentID}(Enrol)$

Rule-based Optimisation

- Can join be executed last? $\rightsquigarrow$ Push select/project before join.

- $\pi_X(R_1 \bowtie R_2) \equiv \pi_{X_1}(R_1) \bowtie \pi_{X_2}(R_2)$, if the join condition involves only attributes in X, how could we derive X_1 and X_2?

  $\pi_{CourseNo,Cname}(Course) \bowtie \pi_{CourseNo,StudentID}(Enrol)$

COURSE		
CourseNo	Cname	Unit
COMP2400	Relational Databases	6
BUSN2011	Management Accounting	6

Rule-based Optimisation

- Can join be executed last? $\rightsquigarrow$ Push select/project before join.
  - $\pi_X(R_1 \bowtie R_2) \equiv \pi_{X_1}(R_1) \bowtie \pi_{X_2}(R_2)$, if the join condition involves only attributes in X, how could we derive X_1 and X_2?

    $\pi_{CourseNo,Cname}(Course) \bowtie \pi_{CourseNo,StudentID}(Enrol)$

COURSE		
CourseNo	Cname	Unit
COMP2400	Relational Databases	6
BUSN2011	Management Accounting	6

$\pi_{CourseNo,Cname}$COURSE	
CourseNo	Cname
COMP2400	Relational Databases
BUSN2011	Management Accounting

Rule-based Optimisation

- Can join be executed last? $\rightsquigarrow$ Push select/project before join.
  - $\pi_X(R_1 \bowtie R_2) \equiv \pi_{X_1}(R_1) \bowtie \pi_{X_2}(R_2)$, if the join condition involves only attributes in X, how could we derive X_1 and X_2?

    $\pi_{CourseNo,Cname}(Course) \bowtie \pi_{CourseNo,StudentID}(Enrol)$

COURSE		
CourseNo	Cname	Unit
COMP2400	Relational Databases	6
BUSN2011	Management Accounting	6

ENROL			
StudentID	CourseNo	Semester	Status
111	BUSN2011	2016 S1	active
222	COMP2400	2016 S1	active
111	COMP2400	2016 S2	active

$\pi_{CourseNo,Cname}$ COURSE	
CourseNo	Cname
COMP2400	Relational Databases
BUSN2011	Management Accounting

Rule-based Optimisation

- Can join be executed last? $\rightsquigarrow$ Push select/project before join.
  - $\pi_X(R_1 \bowtie R_2) \equiv \pi_{X_1}(R_1) \bowtie \pi_{X_2}(R_2)$, if the join condition involves only attributes in X, how could we derive X_1 and X_2?

    $\pi_{CourseNo,Cname}(Course) \bowtie \pi_{CourseNo,StudentID}(Enrol)$

COURSE		
CourseNo	Cname	Unit
COMP2400	Relational Databases	6
BUSN2011	Management Accounting	6

ENROL			
StudentID	CourseNo	Semester	Status
111	BUSN2011	2016 S1	active
222	COMP2400	2016 S1	active
111	COMP2400	2016 S2	active

$\pi_{CourseNo,Cname}$COURSE	
CourseNo	Cname
COMP2400	Relational Databases
BUSN2011	Management Accounting

$\pi_{CourseNo,StudentID}$ENROL	
StudentID	CourseNo
111	BUSN2011
222	COMP2400
111	COMP2400

Rule-based Optimisation

- Can join be executed last? $\rightsquigarrow$ Push select/project before join.
  - $\pi_X(R_1 \bowtie R_2) \equiv \pi_{X_1}(R_1) \bowtie \pi_{X_2}(R_2)$, if the join condition involves only attributes in X, how could we derive X_1 and X_2?

$\pi_{CourseNo,Cname}(Course) \bowtie \pi_{CourseNo,StudentID}(Enrol)$

COURSE

CourseNo	Cname	Unit
COMP2400	Relational Databases	6
BUSN2011	Management Accounting	6

ENROL

StudentID	CourseNo	Semester	Status
111	BUSN2011	2016 S1	active
222	COMP2400	2016 S1	active
111	COMP2400	2016 S2	active

$\pi_{CourseNo,Cname}$COURSE

CourseNo	Cname
COMP2400	Relational Databases
BUSN2011	Management Accounting

$\pi_{CourseNo,StudentID}$ENROL

StudentID	CourseNo
111	BUSN2011
222	COMP2400
111	COMP2400

CourseNo	Cname	StudentID
COMP2400	Relational Databases	222
COMP2400	Relational Databases	111
BUSN2011	Management Accounting	111

Rule-based Optimisation

- Can join be executed last? $\hookrightarrow$ Push select/project before join.

Rule-based Optimisation

- Can join be executed last? $\leadsto$ Push select/project before join.
  - $\pi_X(R_1 \bowtie R_2) \equiv \pi_X(\pi_{X_1}(R_1) \bowtie \pi_{X_2}(R_2))$, if the join condition involves attributes outside X, how could we derive X_1 and X_2?

Rule-based Optimisation

- Can join be executed last? $\leadsto$ Push select/project before join.
  - $\pi_X(R_1 \bowtie R_2) \equiv \pi_X(\pi_{X_1}(R_1) \bowtie \pi_{X_2}(R_2))$, if the join condition involves attributes outside X, how could we derive X_1 and X_2?

    $\pi_{Cname,StudentID}(Course \bowtie Enrol)$

Rule-based Optimisation

- Can join be executed last? $\rightsquigarrow$ Push select/project before join.

  - $\pi_X(R_1 \bowtie R_2) \equiv \pi_X(\pi_{X_1}(R_1) \bowtie \pi_{X_2}(R_2))$, if the join condition involves attributes outside X, how could we derive X_1 and X_2?

    $\pi_{Cname,StudentID}(Course \bowtie Enrol)$

COURSE		
CourseNo	Cname	Unit
COMP2400	Relational Databases	6
BUSN2011	Management Accounting	6

Rule-based Optimisation

- Can join be executed last? $\leadsto$ Push select/project before join.
  - $\pi_X(R_1 \bowtie R_2) \equiv \pi_X(\pi_{X_1}(R_1) \bowtie \pi_{X_2}(R_2))$, if the join condition involves attributes outside X, how could we derive X_1 and X_2?

    $\pi_{Cname,StudentID}(Course \bowtie Enrol)$

COURSE		
CourseNo	Cname	Unit
COMP2400	Relational Databases	6
BUSN2011	Management Accounting	6

ENROL			
StudentID	CourseNo	Semester	Status
111	BUSN2011	2016 S1	active
222	COMP2400	2016 S1	active
111	COMP2400	2016 S2	active

Rule-based Optimisation

- Can join be executed last? $\rightsquigarrow$ Push select/project before join.
  - $\pi_X(R_1 \bowtie R_2) \equiv \pi_X(\pi_{X_1}(R_1) \bowtie \pi_{X_2}(R_2))$, if the join condition involves attributes outside X, how could we derive X_1 and X_2?

    $\pi_{Cname,StudentID}(Course \bowtie Enrol)$

COURSE

CourseNo	Cname	Unit
COMP2400	Relational Databases	6
BUSN2011	Management Accounting	6

ENROL

StudentID	CourseNo	Semester	Status
111	BUSN2011	2016 S1	active
222	COMP2400	2016 S1	active
111	COMP2400	2016 S2	active

CourseNo	Cname	Unit	StudentID	Semester	Status
COMP2400	Relational Databases	6	222	2016 S1	active
COMP2400	Relational Databases	6	111	2016 S2	active
BUSN2011	Management Accounting	6	111	2016 S1	active

Rule-based Optimisation

- Can join be executed last? ⇝ Push select/project before join.
  - $\pi_X(R_1 \bowtie R_2) \equiv \pi_X(\pi_{X_1}(R_1) \bowtie \pi_{X_2}(R_2))$, if the join condition involves attributes outside X, how could we derive X_1 and X_2?

    $\pi_{Cname,StudentID}(Course \bowtie Enrol)$

COURSE		
CourseNo	Cname	Unit
COMP2400	Relational Databases	6
BUSN2011	Management Accounting	6

ENROL			
StudentID	CourseNo	Semester	Status
111	BUSN2011	2016 S1	active
222	COMP2400	2016 S1	active
111	COMP2400	2016 S2	active

CourseNo	Cname	Unit	StudentID	Semester	Status
COMP2400	Relational Databases	6	222	2016 S1	active
COMP2400	Relational Databases	6	111	2016 S2	active
BUSN2011	Management Accounting	6	111	2016 S1	active

Cname	StudentID
Relational Databases	222
Relational Databases	111
Management Accounting	111

Rule-based Optimisation

- Can join be executed last? $\hookrightarrow$ Push select/project before join
  - $\pi_X(R_1 \bowtie R_2) \equiv \pi_X(\pi_{X_1}(R_1) \bowtie \pi_{X_2}(R_2))$, if the join condition involves attributes outside X, how could we derive X_1 and X_2?
    $\pi_{Cname,StudentID}(Course \bowtie Enrol) = \pi_{Cname}(Course) \bowtie \pi_{StudentID}(Enrol)$?

Rule-based Optimisation

- Can join be executed last? $\hookrightarrow$ Push select/project before join
  - $\pi_X(R_1 \bowtie R_2) \equiv \pi_X(\pi_{X_1}(R_1) \bowtie \pi_{X_2}(R_2))$, if the join condition involves attributes outside X, how could we derive X_1 and X_2?

  $\pi_{Cname,StudentID}(Course \bowtie Enrol) = \pi_{Cname}(Course) \bowtie \pi_{StudentID}(Enrol)$?

COURSE		
CourseNo	Cname	Unit
COMP2400	Relational Databases	6
BUSN2011	Management Accounting	6

Rule-based Optimisation

- Can join be executed last? $\hookrightarrow$ Push select/project before join
  - $\pi_X(R_1 \bowtie R_2) \equiv \pi_X(\pi_{X_1}(R_1) \bowtie \pi_{X_2}(R_2))$, if the join condition involves attributes outside X, how could we derive X_1 and X_2?

$\pi_{Cname,StudentID}(Course \bowtie Enrol) = \pi_{Cname}(Course) \bowtie \pi_{StudentID}(Enrol)$?

COURSE		
CourseNo	Cname	Unit
COMP2400	Relational Databases	6
BUSN2011	Management Accounting	6

π_{Cname}COURSE
Cname
Relational
Management

Rule-based Optimisation

- Can join be executed last? $\hookrightarrow$ Push select/project before join
  - $\pi_X(R_1 \bowtie R_2) \equiv \pi_X(\pi_{X_1}(R_1) \bowtie \pi_{X_2}(R_2))$, if the join condition involves attributes outside X, how could we derive X_1 and X_2?

$\pi_{Cname,StudentID}(Course \bowtie Enrol) = \pi_{Cname}(Course) \bowtie \pi_{StudentID}(Enrol)$?

COURSE

CourseNo	Cname	Unit
COMP2400	Relational Databases	6
BUSN2011	Management Accounting	6

ENROL

StudentID	CourseNo	Semester	Status
111	BUSN2011	2016 S1	active
222	COMP2400	2016 S1	active
111	COMP2400	2016 S2	active

π_{Cname}COURSE

Cname
Relational
Management

Rule-based Optimisation

- Can join be executed last? $\hookrightarrow$ Push select/project before join
  - $\pi_X(R_1 \bowtie R_2) \equiv \pi_X(\pi_{X_1}(R_1) \bowtie \pi_{X_2}(R_2))$, if the join condition involves attributes outside X, how could we derive X_1 and X_2?

$\pi_{Cname,StudentID}(Course \bowtie Enrol) = \pi_{Cname}(Course) \bowtie \pi_{StudentID}(Enrol)$?

COURSE		
CourseNo	Cname	Unit
COMP2400	Relational Databases	6
BUSN2011	Management Accounting	6

ENROL			
StudentID	CourseNo	Semester	Status
111	BUSN2011	2016 S1	active
222	COMP2400	2016 S1	active
111	COMP2400	2016 S2	active

π_{Cname}COURSE
Cname
Relational
Management

$\pi_{StudentID}$ENROL
StudentID
111
222

Rule-based Optimisation

- Can join be executed last? $\hookrightarrow$ Push select/project before join
  - $\pi_X(R_1 \bowtie R_2) \equiv \pi_X(\pi_{X_1}(R_1) \bowtie \pi_{X_2}(R_2))$, if the join condition involves attributes outside X, how could we derive X_1 and X_2?

$\pi_{Cname,StudentID}(Course \bowtie Enrol) = \pi_{Cname}(Course) \bowtie \pi_{StudentID}(Enrol)$?

COURSE		
CourseNo	Cname	Unit
COMP2400	Relational Databases	6
BUSN2011	Management Accounting	6

ENROL			
StudentID	CourseNo	Semester	Status
111	BUSN2011	2016 S1	active
222	COMP2400	2016 S1	active
111	COMP2400	2016 S2	active

π_{Cname}COURSE
Cname
Relational
Management

$\pi_{StudentID}$ENROL
StudentID
111
222

Is $\pi_{Cname}(Course) \bowtie \pi_{StudentID}(Enrol)$ our desired result?

Rule-based Optimisation

- Can join be executed last? $\hookrightarrow$ Push select/project before join
  - $\pi_X(R_1 \bowtie R_2) \equiv \pi_X(\pi_{X_1}(R_1) \bowtie \pi_{X_2}(R_2))$, if the join condition involves attributes outside X, how could we derive X_1 and X_2?

$\pi_{Cname,StudentID}(Course \bowtie Enrol) = \pi_{Cname}(Course) \bowtie \pi_{StudentID}(Enrol)$?

COURSE

CourseNo	Cname	Unit
COMP2400	Relational Databases	6
BUSN2011	Management Accounting	6

ENROL

StudentID	CourseNo	Semester	Status
111	BUSN2011	2016 S1	active
222	COMP2400	2016 S1	active
111	COMP2400	2016 S2	active

π_{Cname}COURSE

Cname
Relational
Management

$\pi_{StudentID}$ENROL

StudentID
111
222

Is $\pi_{Cname}(Course) \bowtie \pi_{StudentID}(Enrol)$ our desired result?

No. $\pi_{Cname,StudentID}(Course \bowtie Enrol) \neq \pi_{Cname}(Course) \bowtie \pi_{StudentID}(Enrol)$

Rule-based Optimisation

- Can join be executed last? $\hookrightarrow$ Push select/project before join.

- $\pi_X(R_1 \bowtie R_2) = \pi_X(\pi_{X_1}(R_1) \bowtie \pi_{X_2}(R_2))$, if the join condition involves attributes outside X, how could we derive X_1 and X_2?

  $\pi_{Cname,StudentID}(Course \bowtie Enrol)$

  $\pi_{Cname,StudentID}(\pi_{CourseNo,Cname}(Course) \bowtie \pi_{CourseNo,StudentID}(Enrol))$

Rule-based Optimisation

- Can join be executed last? $\hookrightarrow$ Push select/project before join.

- $\pi_X(R_1 \bowtie R_2) = \pi_X(\pi_{X_1}(R_1) \bowtie \pi_{X_2}(R_2))$, if the join condition involves attributes outside X, how could we derive X_1 and X_2?

  $\pi_{Cname,StudentID}(Course \bowtie Enrol)$

  $\pi_{Cname,StudentID}(\pi_{CourseNo,Cname}(Course) \bowtie \pi_{CourseNo,StudentID}(Enrol))$

COURSE		
CourseNo	Cname	Unit
COMP2400	Relational Databases	6
BUSN2011	Management Accounting	6

Rule-based Optimisation

- Can join be executed last? $\hookrightarrow$ Push select/project before join.

- $\pi_X(R_1 \bowtie R_2) = \pi_X(\pi_{X_1}(R_1) \bowtie \pi_{X_2}(R_2))$, if the join condition involves attributes outside X, how could we derive X_1 and X_2?

  $\pi_{Cname,StudentID}(Course \bowtie Enrol)$

  $\pi_{Cname,StudentID}(\pi_{CourseNo,Cname}(Course) \bowtie \pi_{CourseNo,StudentID}(Enrol))$

COURSE		
CourseNo	Cname	Unit
COMP2400	Relational Databases	6
BUSN2011	Management Accounting	6

$\pi_{CourseNo,Cname}$ COURSE	
CourseNo	Cname
COMP2400	Relational Databases
BUSN2011	Management Accounting

Rule-based Optimisation

- Can join be executed last? $\hookrightarrow$ Push select/project before join.

- $\pi_X(R_1 \bowtie R_2) = \pi_X(\pi_{X_1}(R_1) \bowtie \pi_{X_2}(R_2))$, if the join condition involves attributes outside X, how could we derive X_1 and X_2?

  $\pi_{Cname,StudentID}(Course \bowtie Enrol)$

  $\pi_{Cname,StudentID}(\pi_{CourseNo,Cname}(Course) \bowtie \pi_{CourseNo,StudentID}(Enrol))$

COURSE		
CourseNo	Cname	Unit
COMP2400	Relational Databases	6
BUSN2011	Management Accounting	6

ENROL			
StudentID	CourseNo	Semester	Status
111	BUSN2011	2016 S1	active
222	COMP2400	2016 S1	active
111	COMP2400	2016 S2	active

$\pi_{CourseNo,Cname}$ COURSE	
CourseNo	Cname
COMP2400	Relational Databases
BUSN2011	Management Accounting

Rule-based Optimisation

- Can join be executed last? $\hookrightarrow$ Push select/project before join.

- $\pi_X(R_1 \bowtie R_2) = \pi_X(\pi_{X_1}(R_1) \bowtie \pi_{X_2}(R_2))$, if the join condition involves attributes outside X, how could we derive X_1 and X_2?

  $\pi_{Cname,StudentID}(Course \bowtie Enrol)$

  $\pi_{Cname,StudentID}(\pi_{CourseNo,Cname}(Course) \bowtie \pi_{CourseNo,StudentID}(Enrol))$

COURSE		
CourseNo	Cname	Unit
COMP2400	Relational Databases	6
BUSN2011	Management Accounting	6

ENROL			
StudentID	CourseNo	Semester	Status
111	BUSN2011	2016 S1	active
222	COMP2400	2016 S1	active
111	COMP2400	2016 S2	active

$\pi_{CourseNo,Cname}$ COURSE	
CourseNo	Cname
COMP2400	Relational Databases
BUSN2011	Management Accounting

$\pi_{CourseNo,StudentID}$ ENROL	
StudentID	CourseNo
111	BUSN2011
222	COMP2400
111	COMP2400

Australian National University

Rule-based Optimisation

- Can join be executed last? $\hookrightarrow$ Push select/project before join.
  - $\pi_X(R_1 \bowtie R_2) = \pi_X(\pi_{X_1}(R_1) \bowtie \pi_{X_2}(R_2))$, if the join condition involves attributes outside X, how could we derive X_1 and X_2?

    $\pi_{Cname,StudentID}(Course \bowtie Enrol)$

    $\pi_{Cname,StudentID}(\pi_{CourseNo,Cname}(Course) \bowtie \pi_{CourseNo,StudentID}(Enrol))$

COURSE

CourseNo	Cname	Unit
COMP2400	Relational Databases	6
BUSN2011	Management Accounting	6

ENROL

StudentID	CourseNo	Semester	Status
111	BUSN2011	2016 S1	active
222	COMP2400	2016 S1	active
111	COMP2400	2016 S2	active

$\pi_{CourseNo,Cname}$ COURSE

CourseNo	Cname
COMP2400	Relational Databases
BUSN2011	Management Accounting

$\pi_{CourseNo,StudentID}$ ENROL

StudentID	CourseNo
111	BUSN2011
222	COMP2400
111	COMP2400

CourseNo	Cname	StudentID
COMP2400	Relational Databases	222
COMP2400	Relational Databases	111
BUSN2011	Management Accounting	111

Rule-based Optimisation

- Can join be executed last? $\hookrightarrow$ Push select/project before join.

- $\pi_X(R_1 \bowtie R_2) = \pi_X(\pi_{X_1}(R_1) \bowtie \pi_{X_2}(R_2))$; if the join condition involves attributes outside X, how could we derive X_1 and X_2?

  $\pi_{Cname,StudentID}(Course \bowtie Enrol)$

  $\pi_{Cname,StudentID}(\pi_{CourseNo,Cname}(Course) \bowtie \pi_{CourseNo,StudentID}(Enrol))$

COURSE

CourseNo	Cname	Unit
COMP2400	Relational Databases	6
BUSN2011	Management Accounting	6

ENROL

StudentID	CourseNo	Semester	Status
111	BUSN2011	2016 S1	active
222	COMP2400	2016 S1	active
111	COMP2400	2016 S2	active

$\pi_{CourseNo,Cname}$COURSE

CourseNo	Cname
COMP2400	Relational Databases
BUSN2011	Management Accounting

$\pi_{CourseNo,StudentID}$ENROL

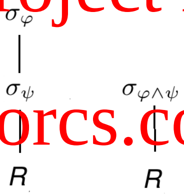
StudentID	CourseNo
111	BUSN2011
222	COMP2400
111	COMP2400

CourseNo	Cname	StudentID
COMP2400	Relational Databases	222
COMP2400	Relational Databases	111
BUSN2011	Management Accounting	111

Cname	StudentID
Relational Databases	222
Relational Databases	111
Management Accounting	111

Heuristic Rules and Query Trees

- (1) $\sigma_{\varphi}(\sigma_{\psi}(R)) \equiv \sigma_{\varphi \wedge \psi}(R)$:



Heuristic Rules and Query Trees

(1) $\sigma_{\varphi}(\sigma_{\psi}(R)) \equiv \sigma_{\varphi \wedge \psi}(R)$;

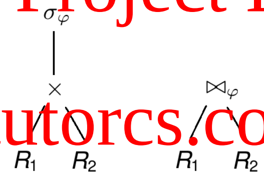
$$\begin{tikzcd}[row sep=large] \sigma_{\varphi} \arrow[d, dash] & \\ \sigma_{\psi} \arrow[d, dash] & \sigma_{\varphi \wedge \psi} \arrow[d, dash] \\ R & R \end{tikzcd}$$

(2) $\pi_X(\pi_Y(R)) \equiv \pi_X(R)$ if $X \subseteq Y$;

$$\begin{tikzcd}[row sep=large] \pi_X \arrow[d, dash] & \text{if } X \subseteq Y \\ \pi_Y \arrow[d, dash] & \pi_X \arrow[d, dash] \\ R & R \end{tikzcd}$$

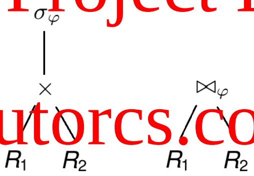
Heuristic Rules

(3) $\sigma_{\varphi}(R_1 \times R_2) \equiv R_1 \bowtie_{\varphi} R_2$

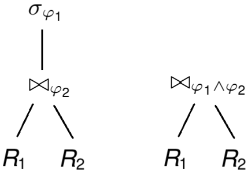


Heuristic Rules

(3) $\sigma_{\varphi}(R_1 \times R_2) \equiv R_1 \bowtie_{\varphi} R_2$



(4) $\sigma_{\varphi_1}(R_1 \bowtie_{\varphi_2} R_2) \equiv R_1 \bowtie_{\varphi_1 \wedge \varphi_2} R_1$



Push-down Selection – Example

- Given the relation schemas:

  PERSON(id, first_name, last_name, year_born)

  MOVIE(title, production_year, country, run_time, major_genre)

  ROLE(id, mtitle, mprod_year, description, credits)

- **Query: List all war movies that are performed by 'Tom Cruise'.**

$$\pi_{title,production\_year}(\sigma_{title=mtitle \wedge production\_year=mprod\_year}(\sigma_{major\_genre='war' \wedge first\_name='Tom' \wedge last\_name='Cruise'}(\text{MOVIE} \times (\text{PERSON} \bowtie \text{ROLE}))))$$

Push-down Selection – Example

- Given the relation schemas:

  PERSON(id, first_name, last_name, year_born)

  MOVIE(title, production_year, country, run_time, major_genre)

  ROLE(id, mtitle, mprod_year, description, credits)

- **Query: List all war movies that are performed by 'Tom Cruise'.**

  $\pi_{title,production\_year}(\sigma_{title=mtitle \wedge production\_year=mprod\_year}(\sigma_{major\_genre='war' \wedge first\_name='Tom' \wedge last\_name='Cruise'}(\text{MOVIE} \times (\text{PERSON} \bowtie \text{ROLE}))))$

- **Question**: Can we apply the following rule to optimise the query?
  $\sigma_{\varphi_1 \wedge \varphi_2}(R_1 \times R_2) \equiv \sigma_{\varphi_1}(R_1) \times \sigma_{\varphi_2}(R_2)$ if φ_1 contains only attributes in R_1 and φ_2 contains only attributes in R_2

Push-down Selection – Example

- Given the relation schemas:

  PERSON(id, first_name, last_name, year_born)

  MOVIE(title, production_year, country, run_time, major_genre)

  ROLE(id, mtitle, mprod_year, description, credits)

- **Query: List all war movies that are performed by 'Tom Cruise'.**

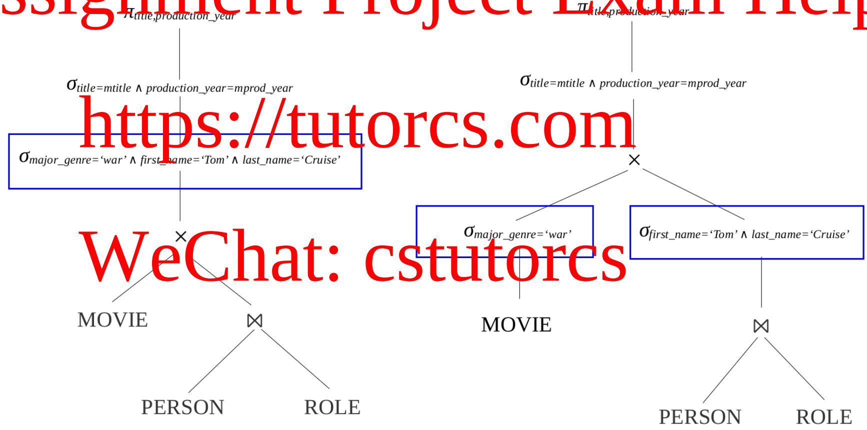
  $\pi_{title,production\_year}(\sigma_{title=mtitle \wedge production\_year=mprod\_year}(\sigma_{major\_genre=\text{'war'} \wedge first\_name=\text{'Tom'} \wedge last\_name=\text{'Cruise'}}(\text{MOVIE} \times (\text{PERSON} \bowtie \text{ROLE}))))$

- **Question**: Can we apply the following rule to optimise the query? $\sigma_{\varphi_1 \wedge \varphi_2}(R_1 \times R_2) \equiv \sigma_{\varphi_1}(R_1) \times \sigma_{\varphi_2}(R_2)$ if φ_1 contains only attributes in R_1 and φ_2 contains only attributes in R_2
- We would have

  $\pi_{title,production\_year}(\sigma_{title=mtitle \wedge production\_year=mprod\_year}(\sigma_{major\_genre=\text{'war'}}(\text{MOVIE})$

  $\times \sigma_{first\_name=\text{'Tom'} \wedge last\_name=\text{'Cruise'}}(\text{PERSON} \bowtie \text{ROLE})))$

Push-down Selection – Example



Push-down Selection – Example

- Given the relation schemas:

  PERSON(id, first_name, last_name, year_born)

  MOVIE(title, production_year, country, run_time, major_genre)

  ROLE(id, mtitle, mprod_year, description, credits)

- **Query: List all war movies that are performed by 'Tom Cruise'.**

$$\pi_{title,production\_year}(\sigma_{title=mtitle \wedge production\_year=mprod\_year}(\sigma_{major\_genre=\text{'war'}}(\text{MOVIE})$$

$$\times \sigma_{first\_name=\text{'Tom'} \wedge last\_name=\text{'Cruise'}}(\text{PERSON} \bowtie \text{ROLE})))$$

Push-down Selection – Example

- Given the relation schemas:

  PERSON(id, first_name, last_name, year_born)

  MOVIE(title, production_year, country, run_time, major_genre)

  ROLE(id, mtitle, mprod_year, description, credits)

- **Query: List all war movies that are performed by 'Tom Cruise'.**

$$\pi_{title,production\_year}(\sigma_{title=mtitle \wedge production\_year=mprod\_year}(\sigma_{major\_genre='war'}(\text{MOVIE})$$
$$\times \sigma_{first\_name='Tom' \wedge last\_name='Cruise'}(\text{PERSON} \bowtie \text{ROLE})))$$

- Can we apply $\sigma_\varphi(R_1 \times R_2) \equiv R_1 \bowtie_\varphi R_2$?

Push-down Selection – Example

- Given the relation schemas:
  PERSON(id, first_name, last_name, year_born)
  MOVIE(title, production_year, country, run_time, major_genre)
  ROLE(id, mtitle, mprod_year, description, credits)

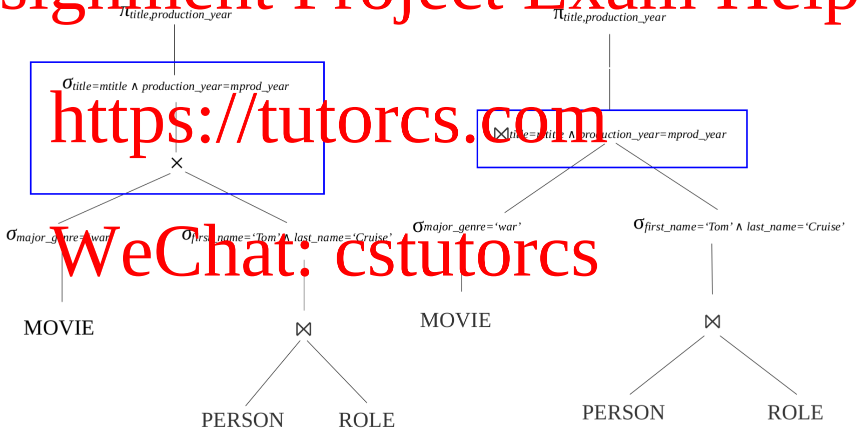
- **Query: List all war movies that are performed by 'Tom Cruise'.**

$$\pi_{title,production\_year}(\sigma_{title=mtitle \wedge production\_year=mprod\_year}(\sigma_{major\_genre=\text{'war'}}(\text{MOVIE}) \times \sigma_{first\_name=\text{'Tom'} \wedge last\_name=\text{'Cruise'}}(\text{PERSON} \bowtie \text{ROLE})))$$

- Can we apply $\sigma_{\varphi}(R_1 \times R_2) \equiv R_1 \bowtie_{\varphi} R_2$?
- We would have

$$\pi_{title,production\_year}(\sigma_{major\_genre=\text{'war'}}(\text{MOVIE}) \bowtie_{title=mtitle \wedge production\_year=mprod\_year} (\sigma_{first\_name=\text{'Tom'} \wedge last\_name=\text{'Cruise'}}(\text{PERSON} \bowtie \text{ROLE})))$$

Push-down Selection – Example



Push-down Projection – Example

- Given the relation schemas:

  PERSON(id, first_name, last_name, year_born)

  MOVIE(title, production_year, country, run_time, major_genre)

  ROLE(id, mtitle, mprod_year, description, credits)

- **Query:** **List all war movies that are performed by 'Tom Cruise'.**

$$\pi_{title,production\_year}(\sigma_{major\_genre='war'}(\text{MOVIE}) \bowtie_{title=mtitle \land production\_year=mprod\_year} ($$
$$\sigma_{first\_name='Tom' \land last\_name='Cruise'}(\text{PERSON} \bowtie \text{ROLE})))$$

Push-down Projection – Example

- Given the relation schemas:

  PERSON(id, first_name, last_name, year_born)

  MOVIE(title, production_year, country, run_time, major_genre)

  ROLE(id, mtitle, mprod_year, description, credits)

- **Query: List all war movies that are performed by 'Tom Cruise'.**

$\pi_{title,production\_year}(\sigma_{major\_genre=\text{'war'}}(\text{MOVIE}) \bowtie_{title=mtitle \wedge production\_year=mprod\_year} ($
$\sigma_{first\_name=\text{'Tom'} \wedge last\_name=\text{'Cruise'}}(\text{PERSON} \bowtie \text{ROLE})))$

- **Question**: Can we apply the following rule to optimise the query?

$$\pi_X(R_1 \bowtie R_2) \equiv \pi_X(\pi_{X_1}(R_1) \bowtie \pi_{X_2}(R_2)),$$

where X_i contains attributes both in R_i and X, and ones both in R_1 and R_2

Push-down Projection – Example

- Given the relation schemas:

  PERSON(id, first_name, last_name, year_born)
  MOVIE(title, production_year, country, run_time, major_genre)
  ROLE(id, mtitle, mprod_year, description, credits)

- **Query: List all war movies that are performed by 'Tom Cruise'.**

$\pi_{title,production\_year}(\sigma_{major\_genre='war'}(\text{MOVIE}) \bowtie_{title=mtitle \wedge production\_year=mprod\_year} ($
$\sigma_{first\_name='Tom' \wedge last\_name='Cruise'}(\text{PERSON} \bowtie \text{ROLE})))$

Push-down Projection – Example

- Given the relation schemas:

  PERSON(id, first_name, last_name, year_born)

  MOVIE(title, production_year, country, run_time, major_genre)

  ROLE(id, mtitle, mprod_year, description, credits)

- **Query: List all war movies that are performed by 'Tom Cruise'.**

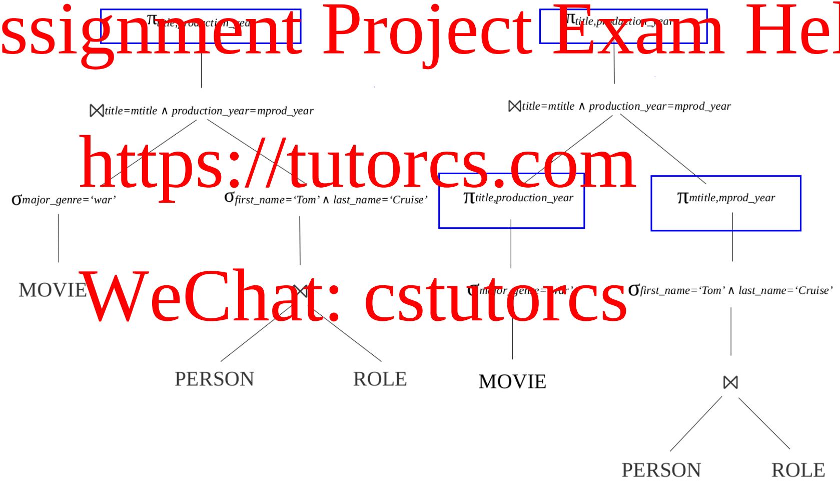
$$\pi_{title,production\_year}(\sigma_{major\_genre='war'}(\text{MOVIE}) \bowtie_{title=mtitle \wedge production\_year=mprod\_year} ($$
$$\sigma_{first\_name='Tom' \wedge last\_name='Cruise'}(\text{PERSON} \bowtie \text{ROLE})))$$

- We would have:

$$\pi_{title,production\_year}(\pi_{title,production\_year}(\sigma_{major\_genre='war'}(\text{MOVIE}))$$
$$\bowtie_{title=mtitle \wedge production\_year=mprod\_year}$$
$$(\pi_{mtitle,mprod\_year}(\sigma_{first\_name='Tom' \wedge last\_name='Cruise'}(\text{PERSON} \bowtie \text{ROLE}))))$$

We further apply some rules to optimise the query ...

Push-down Projection – Example



Cost-based Optimisation (not assessed)

- Consider CHARTS={Rank, Artist, Song} with 100 tuples and 3 attributes

Rank	Artist	Song
1	Chingy	Right Thurr
2	Scribe	Stand up
3	Aguilera and Kim	Can't hold us down
4	Evanescence	Going under
5	Justin Timberlake	Senorita
6	Brooke Fraser	Better
7	Black Eyed Peas	Where is the love?
...	...	...
...	...	...

- Compare two strategies of evaluating "Who is top of the pops?":
  - $\sigma_{\text{Rank}=1}(\pi_{\text{Rank, Artist}}(\text{CHARTS}))$
  - $\pi_{\text{Rank, Artist}}(\sigma_{\text{Rank}=1}(\text{CHARTS}))$

Cost-based Optimisation (not assessed)

- Consider CHARTS={Rank, Artist, Song} with 100 tuples and 3 attributes

Rank	Artist	Song
1	Chingy	Right Thurr
2	Scribe	Stand up
3	Aguilera and Kim	Can't hold us down
4	Evanescence	Going under
5	Justin Timberlake	Senorita
6	Brooke Fraser	Better
7	Black Eyed Peas	Where is the love?
...	...	...
...	...	...

- Compare two strategies of evaluating "Who is top of the pops?":
  - $\sigma_{\text{Rank}=1}(\pi_{\text{Rank, Artist}}(\text{CHARTS}))$
  - $\pi_{\text{Rank, Artist}}(\sigma_{\text{Rank}=1}(\text{CHARTS}))$

  Selection before Projection is preferred.

Cost-based Optimisation (not assessed)

- Consider CHARTS={Rank, Artist, ...} with 100 tuples and 50 attributes:

Rank	Artist	Song	...	...	...
1	Chingy	Right Thurr	...	...	...
2	Scribe	Stand up	...	...	...
3	Aguilera and Kim	Can't hold us down	...	...	...
4	Evanescence	Going under	...	...	...
5	Justin Timberlake	Senorita	...	...	...
6	Brooke Fraser	Better	...	...	...
7	Black Eyed Peas	Where is the love?	..	...	...
...	...	...	...	...	...

- Compare two strategies of evaluating?
  - $\sigma_{\text{Rank} > 10}(\pi_{\text{Rank, Artist}}(\text{CHARTS}))$
  - $\pi_{\text{Rank, Artist}}(\sigma_{\text{Rank} > 10}(\text{CHARTS}))$

Cost-based Optimisation (not assessed)

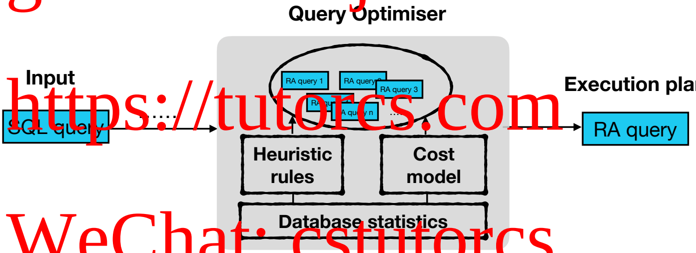
- Consider CHARTS={Rank, Artist ...} with 100 tuples and 50 attributes:

Rank	Artist	Song	...	...	...
1	Chingy	Right Thurr	...	...	...
2	Scribe	Stand up	...	...	...
3	Aguilera and Kim	Can't hold us down	...	...	...
4	Evanescence	Going under	...	...	...
5	Justin Timberlake	Senorita	...	...	...
6	Brooke Fraser	Better	...	...	...
7	Black Eyed Peas	Where is the love?	..	...	...
...	...	...	...	...	...

- Compare two strategies of evaluating?
  - $\sigma_{\text{Rank} > 10}(\pi_{\text{Rank, Artist}}(\text{CHARTS}))$
  - $\pi_{\text{Rank, Artist}}(\sigma_{\text{Rank} > 10}(\text{CHARTS}))$

  Projection before Selection is preferred.

Query Optimisation



- Trade-off:

  Time for executing a RA query vs **Time for finding a better RA query**

(credit cookie) memorising vs understanding