Amr Elhelw's

TECH VAULT

RELATIONAL ALGEBRA DEEP DIVE

Relation

- Represents a kind of entity/object, with certain attributes.

- A set of tuples
  - Unordered
  - No duplicates

- A tuple = one entity/object
  - Specifies the attribute values

`Car(make, model, year, color, transmission)`

make	model	year	color	transmission
Toyota	Camry	2015	red	Auto
Honda	Civic	2012	blue	Manual
Ford	Focus	2015	grey	Auto

Relation

- Values are atomic/scalar
- Primary key
  - Unique
  - Identifies a tuple

`Car(make, model, year, color, transmission)`

make	model	year	color	transmission
Toyota	Camry	2015	red	Auto
Honda	Civic	2012	blue	Manual
Ford	Focus	2015	grey	Auto

Relation

- Values are atomic/scalar
- Primary key
  - Unique
  - Identifies a tuple

Car(id, make, model, year, color, transmission)

id	make	model	year	color	transmission
45	Toyota	Camry	2015	red	Auto
46	Honda	Civic	2012	blue	Manual
47	Ford	Focus	2015	grey	Auto

σ Selection

ρ Rename

π Projection

∪ Union

∩ Intersection

− Difference

× Product

⋈ Join

σ Selection

- Output tuples that satisfy a condition
  - Combine conditions with ∧ (and), ∨ (or)
- Syntax: $\sigma_{condition(s)}(input)$

Car(<u>id</u>, make, year)

id	make	year
45	Toyota	2015
46	Honda	2012
47	Ford	2015
48	Toyota	2023

$\sigma_{make='Toyota'}(Car)$

id	make	year
45	Toyota	2015
48	Toyota	2023

$\sigma_{make='Toyota' \wedge year<2020}(Car)$

id	make	year
45	Toyota	2015

ρ Rename

- Output a relation with renamed attributes
- Syntax: $\rho_{A1, A2, ..., An}(input)$

Car(id, make, year)

id	make	year
45	Toyota	2015
46	Honda	2012
47	Ford	2015
48	Toyota	2023

$\rho_{id,brand,year}(Car)$

id	brand	year
45	Toyota	2015
46	Honda	2012
47	Ford	2015
48	Toyota	2023

π Projection

- Output a relation that has only the specified attributes
- Can add, remove, and change attribute order
- Syntax: $\pi_{A1, A2, ..., An}(input)$

Car(<u>id</u>, make, year)

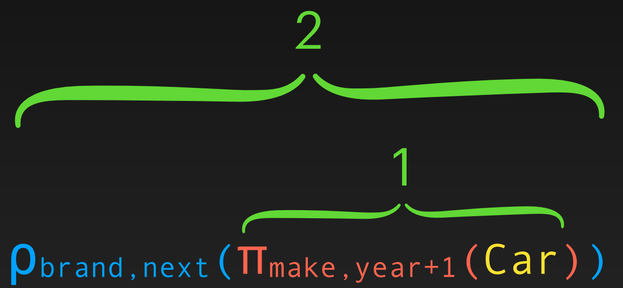
id	make	year
45	Toyota	2015
46	Honda	2012
47	Ford	2015
48	Toyota	2023

$\pi_{year, make}(Car)$

year	make
2015	Toyota
2012	Honda
2015	Ford
2023	Toyota

π Projection

- Output a relation that has only the specified attributes
- Can add, remove, and change attribute order
- Syntax: $\pi_{A1, A2, ..., An}(input)$



Car(id, make, year)

id	make	year
45	Toyota	2015
46	Honda	2012
47	Ford	2015
48	Toyota	2023

$\pi_{make,year+1}(Car)$

make	year+1
Toyota	2016
Honda	2013
Ford	2016
Toyota	2024

brand	next
Toyota	2016
Honda	2013
Ford	2016
Toyota	2024

π Projection

- Output a relation that has only the specified attributes
- Can add, remove, and change attribute order
- Syntax: $\pi_{A1, A2, ..., An}(input)$

Car(id, make, year)

id	make	year
45	Toyota	2015
46	Honda	2012
47	Ford	2015
48	Toyota	2023

$\sigma_{year=2015}(Car)$

id	make	year
45	Toyota	2015
47	Ford	2015

$\pi_{make}(\sigma_{year=2015}(Car))$

make
Toyota
Ford

π Projection

- Output a relation that has only the specified attributes
- Can add, remove, and change attribute order
- Syntax: $\pi_{A1, A2, ..., An}(input)$

Car(<u>id</u>, make, year)

id	make	year
45	Toyota	2015
46	Honda	2012
47	Ford	2015
48	Toyota	2023

$\pi_{make}(Car)$

make
Toyota
Honda
Ford

∪ Union

- Output tuples that appear in one or both of the input relations
- Duplicates are removed
- Inputs must have same attributes
- Syntax: *input1* ∪ *input2*

R(id, name)

id	name
1	Alice
2	Bob
3	Bob

S(id, name)

id	name
5	Tom
3	Jane
2	Bob

R ∪ S

id	name
1	Alice
2	Bob
3	Bob
5	Tom
3	Jane

∪ Union

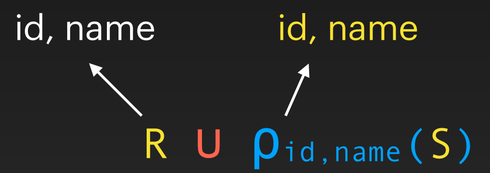
- Output tuples that appear in one or both of the input relations
- Duplicates are removed
- Inputs must have same attributes
- Syntax: *input1* ∪ *input2*

R(id, name)

id	name
1	Alice
2	Bob
3	Bob

S(id, label)

id	label
5	Tom
3	Jane
2	Bob



id	name
1	Alice
2	Bob
3	Bob
5	Tom
3	Jane

∪ Union

- Output tuples that appear in one or both of the input relations
- Duplicates are removed
- Inputs must have same attributes
- Syntax: *input1* ∪ *input2*

R(id, name, country)

id	name	country
1	Alice	US
2	Bob	FR
3	Bob	UK

S(id, name, age)

id	name	age
5	Tom	25
3	Jane	22
2	Bob	36

name
Alice
Bob
Tom
Jane

∩ Intersection

- Output tuples that appear in both of the input relations
- Inputs must have same attributes
- Syntax: *input1* ∩ *input2*

R(id, name)

id	name
1	Alice
2	Bob
3	Bob

S(id, name)

id	name
5	Tom
3	Jane
2	Bob

R ∩ S

id	name
2	Bob

— Difference

- Output tuples that appear in the first but not in the second input relations
- Inputs must have same attributes
- Syntax: *input1 — input2*

R(id, name)

id	name
1	Alice
2	Bob
3	Bob

S(id, name)

id	name
5	Tom
3	Jane
2	Bob

R — S

id	name
1	Alice
3	Bob

Product

- AKA *Cartesian Product* or *Cross Product*
- Output all possible combinations of tuples from the input relations
- Syntax: *input1* × *input2*

R(id, name)

id	name
1	Alice
2	Bob
3	Jane

S(id, loc)

id	loc
5	UK
3	US
2	CA

R × S

R.id	name	S.id	loc
1	Alice	5	UK
1	Alice	3	US
1	Alice	2	CA
2	Bob	5	UK
2	Bob	3	US
2	Bob	2	CA
3	Jane	5	UK
3	Jane	3	US
3	Jane	2	CA

Join

- AKA *inner join*
- Syntax (1): *input1* $\bowtie_{attr(s)}$ *input2*
- Output combinations of tuples from the input relations with matching attribute values
- Join attributes are output only once

R(id, name)

id	name
1	Alice
2	Bob
3	Jane

S(id, loc)

id	loc
5	UK
3	US
2	CA

R $\bowtie_{id}$ S

id	name	loc
2	Bob	CA
3	Jane	US

⋈ Join

- AKA *inner join*
- Syntax (2): *input1* $\bowtie_{condition(s)}$ *input2*
- More general form
- Output combinations of tuples from the input relations that satisfy the join conditions

R(id, name)

id	name
1	Alice
2	Bob
3	Jane

S(id, loc)

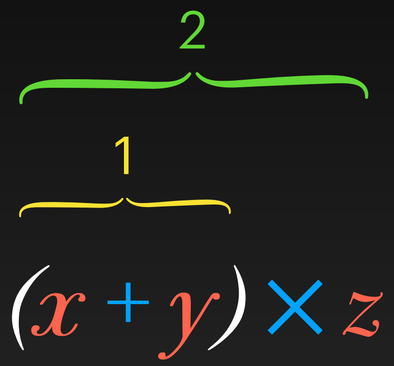
id	loc
5	UK
3	US
2	CA

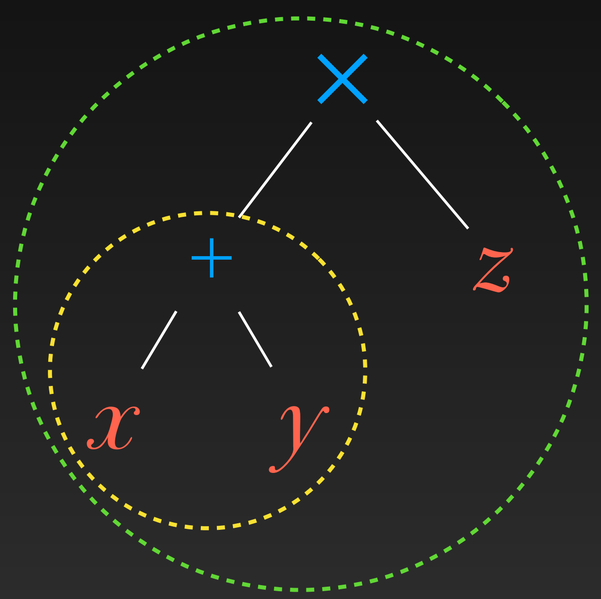
$R \bowtie_{R.id=S.id} S$

R.id	name	S.id	loc
2	Bob	2	CA
3	Jane	3	US

$\sigma_{R.id=S.id}(R \times S)$

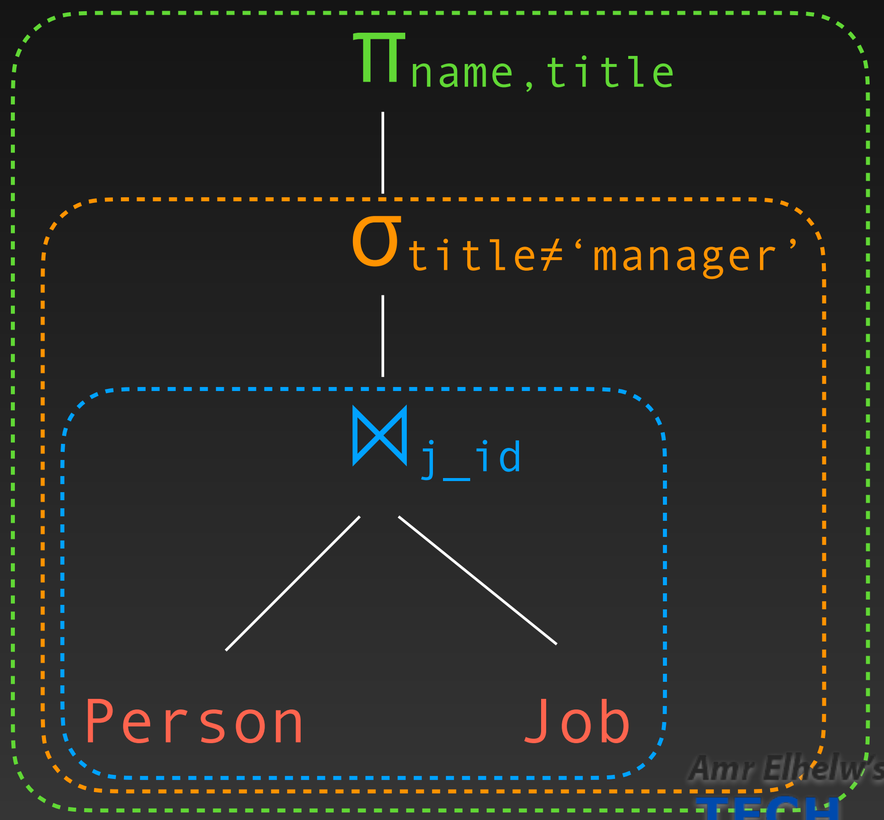
Tree Representation





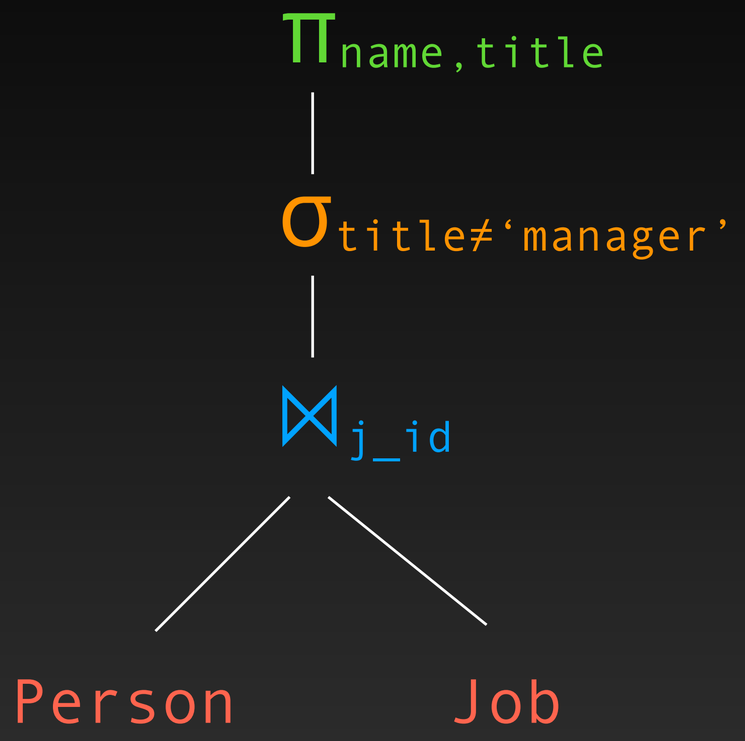
Tree Representation

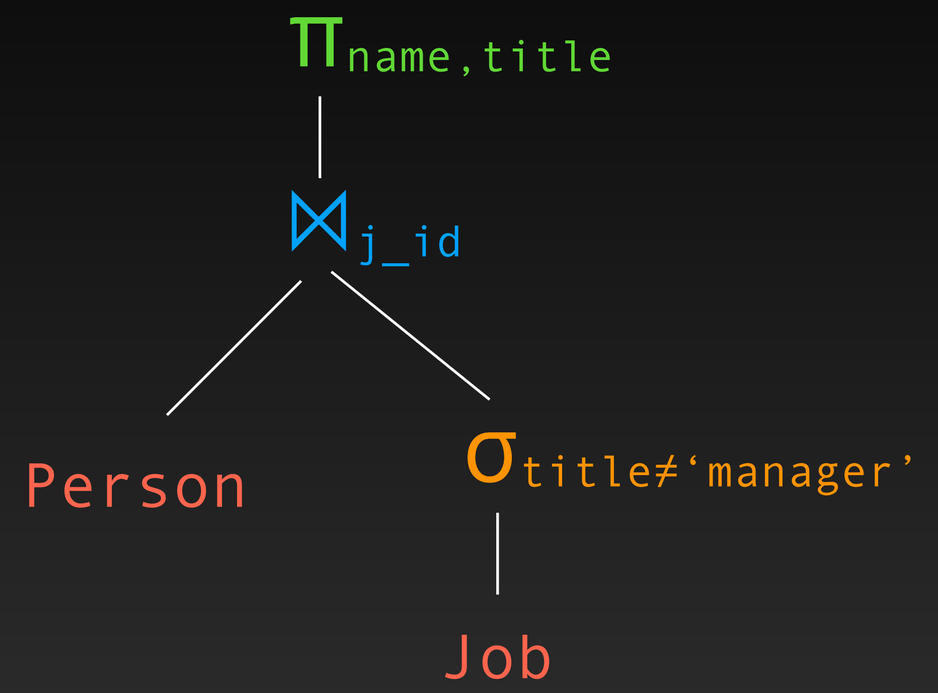
- Relations:
  - `Job(j_id, title)`
  - `Person(p_id, name, j_id)`
- Ask: Return the name and job title for every person who is not a manager



$$\pi_{name,title}(\sigma_{title \neq 'manager'}(Person \bowtie_{j\_id} Job))$$

Order of Operations





$\pi_{name,title}(\sigma_{title \neq 'manager'}(Person \bowtie_{j\_id} Job))$

$\pi_{name,title}(Person \bowtie_{j\_id} \sigma_{title \neq 'manager'}(Job))$