>>

RA Set Operations

❖ RA Set Operations

Relational algebra defines three set operations

- union ... $R \cup S$ ... ($Query_1$) **`UNION`** ($Query_2$)
- intersection ... $R \cap S$ ... ($Query_1$) **`INTERSECT`** ($Query_2$)
- difference ... $R - S$ ... ($Query_1$) **`EXCEPT`** ($Query_2$)

All relations involved must have the same schema (union-compatible)

All operations give a *set* of results (i.e. no duplicates)

To get *bag* semantics, use **`UNION ALL`**, etc.

<< ∧ >>

❖ Union

Union combines two compatible relations into a single relation via set union of sets of tuples.

$r_1 \cup r_2 = \{ t \mid t \in r_1 \vee t \in r_2 \}$, where $r_1(R), r_2(R)$

Result size: $|r_1 \cup r_2| \le |r_1| + |r_2|$ Result schema: R

Algorithmic view:

```
result = r1
for each tuple t in relation r2
    result = result ∪ {t}
```

<< ∧ >>

❖ Intersection

Intersection combines two compatible relations into a single relation via set intersection of sets of tuples.

$r_1 \cap r_2 = \{ t \mid t \in r_1 \wedge t \in r_2 \}$, where $r_1(R), r_2(R)$

Result size: $|r_1 \cap r_2| \leq \min(|r_1|,|r_2|)$ Result schema: R

Algorithmic view:

```
result = {}
for each tuple t in relation r1
    if (t ∈ r2) { result = result ∪ {t} }
```

<< ∧ >>

❖ Intersection (cont)

Examples of union and intersection:

T = Sel[B=1](R)

A	B	C	D
a	1	x	4
e	1	y	4

U = Sel[C=x](R)

A	B	C	D
a	1	x	4
d	8	x	5

T union U

A	B	C	D
a	1	x	4
d	8	x	5
e	1	y	4

T intersect U

A	B	C	D
a	1	x	4

<< ^ >>

❖ Intersection (cont)

Querying with relational algebra (set operations)...

- Bars where either John or Gernot drinks

```
JohnBars = Proj[bar](Sel[drinker=John](Frequents))
GernotBars = Proj[bar](Sel[drinker=Gernot](Frequents))

Result = JohnBars union GernotBars
```

- Bars where both John and Gernot drink

```
Result = JohnBars  intersect  GernotBars
```

<< ∧ >>

❖ Difference

Difference finds the set of tuples that exist in one relation but do not occur in a second compatible relation.

$r_1 - r_2 = \{ t \mid t \in r_1 \wedge t \notin r_2 \}$, where $r_1(R), r_2(R)$

Uses same notion of relation compatibility as union.

Note: tuples in r_2 but not r_1 do not appear in the result

- i.e. set difference != complement of set intersection

Algorithmic view:

```
result = {}
for each tuple t in relation r1
    if (!(t ∈ r2)) { result = result ∪ {t} }
```

<< ^ >>

❖ Difference (cont)

Examples of difference:

T = Sel[B=1](R)

A	B	C	D
a	1	x	4
e	1	y	4

U = Sel[C=x](R)

A	B	C	D
a	1	x	4
d	8	x	5

T - U

A	B	C	D
e	1	y	4

U - T

A	B	C	D
d	8	x	5

Clearly, difference is not symmetric.

<< ^

❖ Difference (cont)

Querying with relational algebra (difference) ...

- Bars where John drinks and Gernot doesn't

```
JohnBars = Proj[bar](Sel[drinker=John](Frequents))
GernotBars = Proj[bar](Sel[drinker=Gernot](Frequents))

Result = JohnBars - GernotBars
```

- Bars that sell VB but not New

```
VBBars = Proj[bar](Sel[beer=VB](Sells))
NewBars = Proj[bar](Sel[beer=New](Sells))

Result = VBBars - NewBars
```

Produced: 11 Nov 2020