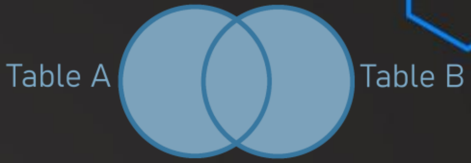
Day 7

UNION

FULL OUTER JOIN



employee	city	sales
Sandra	Frankfurt	500
Sabine	Munich	300
Peter	Hamburg	200
Manuel	Hamburg	400
Michael	Munich	100
Frank	Frankfurt	100

employee	bonus
Sandra	YES
Sabine	YES
Peter	NO
Manuel	YES
Simon	NO

bonus.employee	sales.employee	city	sales	bonus
null	Sandra	Frankfurt	500	YES
null	Sabine	Munich	300	YES
null	Peter	Hamburg	200	NO
null	Manuel	Hamburg	400	YES
null	Michael	Munich	100	*null*
		rankfurt	100	*null*
		ull	*null*	NO

Combining columns

UNION

New York

name	sales
Sandra	500
Maya	300
Peter	200

Delhi

name	sales
Sunita	600
Anil	400
Shanti	100

Combining multiple select statements

UNION

New York

name	sales
Sandra	500
Maya	300
Peter	200

Delhi

name	sales
Sunita	600
Anil	400
Shanti	100

name	sales
Sunita	600
Anil	400
Shanti	100
Sandra	500
Maya	300
Peter	200

SYNTAX

```
SELECT first_name, sales FROM vancouver
UNION
SELECT first_name, sales FROM delhi
```

3 Things to remember!

How columns are matched?

1st thing to remember

New York

name	sales
Sandra	500
Maya	300
Peter	200

Delhi

name	sales
Sunita	600
Anil	400
Shanti	100

Columns are matched by the order!

1st thing to remember

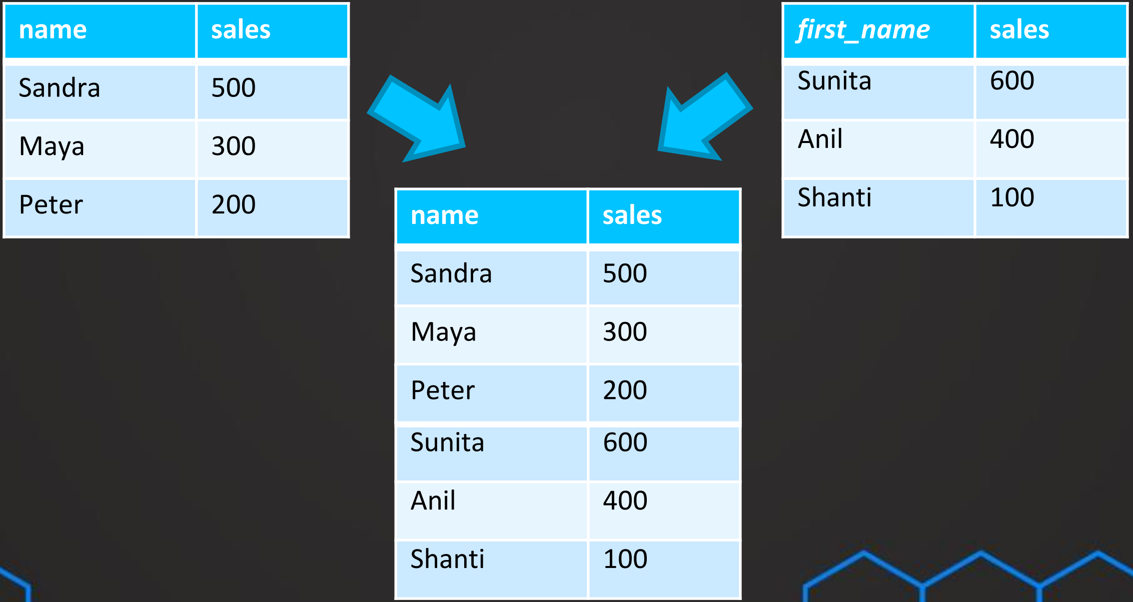
New York

name	sales
Sandra	500
Maya	300
Peter	200

Delhi

first_name	sales
Sunita	600
Anil	400
Shanti	100

UNION



New York

name	sales
Sandra	500
Maya	300
Peter	200

Delhi

first_name	sales
Sunita	600
Anil	400
Shanti	100

name	sales
Sandra	500
Maya	300
Peter	200
Sunita	600
Anil	400
Shanti	100

UNION

New York

name	sales
Sandra	500
Maya	300
Peter	200

Delhi

first_name	sales
Sunita	600
Anil	400
Shanti	100

name	sales
Sandra	500
Maya	300
Peter	200
Sunita	600
Anil	400
Shanti	100

SYNTAX

```
SELECT first_name, sales FROM delhi
UNION
SELECT name, sales FROM vancouver
```

UNION

New York

name	sales
Sandra	500
Maya	300
Peter	200

Delhi

first_name	sales
Sunita	600
Anil	400
Shanti	100

first_name	sales
Sunita	600
Anil	400
Shanti	100
Sandra	500
Maya	300
Peter	200

SYNTAX

```
SELECT first_name, sales FROM delhi
UNION
SELECT first_name, sales FROM vancouver
```

Must be aware of the order in the match

SYNTAX

```
SELECT first_name, sales FROM delhi
UNION
SELECT sales, first_name FROM vancouver
```

UNION

New York

name	sales
Sandra	500
Maya	300
Peter	200

Delhi

first_name	sales
Sunita	600
Anil	400
Shanti	100

first_name	sales
Sunita	600
Anil	400
Shanti	100
500	Sandra
300	Maya
200	Peter

Data type must match!

UNION

New York

name	sales
Sandra	500
Maya	300
Peter	200

Delhi

first_name	sales
Sunita	600
Anil	400
Shanti	100

first_name	sales
Sunita	600
Anil	400
Shanti	100
500	Sandra
300	Maya
200	Peter

No. of columns must match!

Data type must match!

UNION

New York

name	**sales**
Sunita	600
Maya	300
Peter	200

Delhi

first_name	**sales**
Sunita	600
Anil	400
Shanti	100

first_name	**sales**
Sunita	600
Anil	400
Shanti	100
Maya	300
Peter	200

Duplicates are decoupled!

UNION

New York

name	sales
Sunita	600
Maya	300
Peter	200

Delhi

first_name	sales
Sunita	600
Anil	400
Shanti	100

first_name	sales
Sunita	600
Anil	400
Shanti	100
Sunita	600
Maya	300
Peter	200

UNION ALL

SYNTAX

```
SELECT first_name, sales FROM delhi
UNION ALL
SELECT first_name, sales FROM vancouver
```

The order matches the column!

Data types must match!

Duplicates are decoupled!

Correlated subqueries

name	sales
Sunita	600
Anil	400
Shanti	100
Sunita	300
Maya	300
Peter	200
Max	100
Anna	400

name	sales
Sunita	600
Anil	400
Anna	400

Get all people that are above average!

```
SELECT first_name, sales FROM employees
WHERE sales >
      (SELECT AVG(sales) FROM employees)
```

300

Correlated subqueries

name	sales	city
Sunita	600	Delhi
Anil	400	Delhi
Shanti	100	Delhi
Sunita	300	Dallas
Maya	300	Dallas
Peter	200	Dallas
Max	100	Berlin
Anna	400	Berlin

Get all people that are above *average of their city*!

Correlated subquery!

Correlated subqueries

name	sales	city
Sunita	600	Delhi
Anil	400	Delhi
Shanti	100	Delhi
Sunita	300	Dallas
Maya	300	Dallas
Peter	200	Dallas
Max	100	Berlin
Anna	400	Berlin

Get all people that are above *average of their city*!

```
SELECT first_name, sales FROM employees
WHERE sales >
        (…  correlated subquery …)
```

Correlated subqueries

name	sales	city
Sunita	600	Delhi
Anil	400	Delhi
Shanti	100	Delhi
Sunita	300	Dallas
Maya	300	Dallas
Peter	200	Dallas
Max	100	Berlin
Anna	400	Berlin

Get all people that are above *average of their city*!

```
SELECT first_name, sales FROM employees
WHERE sales >
     (SELECT AVG(sales) FROM employees
      … )
```

Correlated subqueries

name	sales	city
Sunita	600	Delhi
Anil	400	Delhi
Shanti	100	Delhi
Sunita	300	Dallas
Maya	300	Dallas
Peter	200	Dallas
Max	100	Berlin
Anna	400	Berlin

Get all people that are above *average of their city*!

```
SELECT first_name, sales FROM employees e1
WHERE sales >
     (SELECT AVG(sales) FROM employees e2
      WHERE e1.city=e2.city )
```

Evaluated for every single row!

Correlated subqueries

name	sales	city
Sunita	600	Delhi
Anil	400	Delhi
Shanti	100	Delhi
Sunita	300	Dallas
Maya	300	Dallas
Peter	200	Dallas
Max	100	Berlin
Anna	400	Berlin

```
WHERE sales >
        (SELECT AVG(sales) FROM employees e2
         WHERE e1.city=e2.city )
```

~366.67

```
SELECT first_name, sales FROM employees e1
WHERE sales >
      (SELECT AVG(sales) FROM employees e2
       WHERE e1.city=e2.city )
```

Correlated subqueries

name	sales	city
Sunita	600	Delhi
Anil	400	Delhi
Shanti	100	Delhi
Sunita	300	Dallas
Maya	300	Dallas
Peter	200	Dallas
Max	100	Berlin
Anna	400	Berlin

```
WHERE sales >
        (SELECT AVG(sales) FROM employees e2
         WHERE e1.city=e2.city )
```

~366.67

```
SELECT first_name, sales FROM employees e1
WHERE sales >
      (SELECT AVG(sales) FROM employees e2
       WHERE e1.city=e2.city )
```

Correlated subqueries

name	sales	city
Sunita	600	Delhi
Anil	400	Delhi
Shanti	100	Delhi
Sunita	300	Dallas
Maya	300	Dallas
Peter	200	Dallas
Max	100	Berlin
Anna	400	Berlin

```
WHERE sales >
        (SELECT AVG(sales) FROM employees e2
         WHERE e1.city=e2.city )
```

~366.67

```
SELECT first_name, sales FROM employees e1
WHERE sales >
     (SELECT AVG(sales) FROM employees e2
      WHERE e1.city=e2.city )
```

Correlated subqueries

name	sales	city
Sunita	600	Delhi
Anil	400	Delhi
~~Shanti~~	~~100~~	~~Delhi~~
Sunita	300	Dallas
Maya	300	Dallas
Peter	200	Dallas
Max	100	Berlin
Anna	400	Berlin

```
WHERE sales >
       (SELECT AVG(sales) FROM employees e2
        WHERE e1.city=e2.city )
```

$$\underbrace{\texttt{(SELECT AVG(sales) FROM employees e2 WHERE e1.city=e2.city)}}_{\sim 366.67}$$

```
SELECT first_name, sales FROM employees e1
WHERE sales >
      (SELECT AVG(sales) FROM employees e2
       WHERE e1.city=e2.city )
```

Correlated subqueries

name	sales	city
Sunita	600	Delhi
Anil	400	Delhi
~~Shanti~~	~~100~~	~~Delhi~~
Sunita	300	Dallas
Maya	300	Dallas
Peter	200	Dallas
Max	100	Berlin
Anna	400	Berlin

```
WHERE sales >
        (SELECT AVG(sales) FROM employees e2
         WHERE e1.city=e2.city )
```

$\underbrace{}_{\sim 266.67}$

```
SELECT first_name, sales FROM employees e1
WHERE sales >
     (SELECT AVG(sales) FROM employees e2
      WHERE e1.city=e2.city )
```

Correlated subqueries

name	sales	city
Sunita	600	Delhi
Anil	400	Delhi
~~Shanti~~	~~100~~	~~Delhi~~
Sunita	300	Dallas
Maya	300	Dallas
~~Peter~~	~~200~~	~~Dallas~~
Max	100	Berlin
Anna	400	Berlin

```
WHERE sales >
        (SELECT AVG(sales) FROM employees e2
         WHERE e1.city=e2.city )
```

~266.67

```
SELECT first_name, sales FROM employees e1
WHERE sales >
     (SELECT AVG(sales) FROM employees e2
      WHERE e1.city=e2.city )
```

Correlated subqueries

name	sales	city
Sunita	600	Delhi
Anil	400	Delhi
~~Shanti~~	~~100~~	~~Delhi~~
Sunita	300	Dallas
Maya	300	Dallas
~~Peter~~	~~200~~	~~Dallas~~
Max	100	Berlin
Anna	400	Berlin

```
WHERE sales >
        (SELECT AVG(sales) FROM employees e2
         WHERE e1.city=e2.city )
```

250

```
SELECT first_name, sales FROM employees e1
WHERE sales >
      (SELECT AVG(sales) FROM employees e2
       WHERE e1.city=e2.city )
```

Correlated subqueries

name	sales	city
Sunita	600	Delhi
Anil	400	Delhi
~~Shanti~~	~~100~~	~~Delhi~~
Sunita	300	Dallas
Maya	300	Dallas
~~Peter~~	~~200~~	~~Dallas~~
~~Max~~	~~100~~	~~Berlin~~
Anna	400	Berlin

```
WHERE sales >
          (SELECT AVG(sales) FROM employees e2
           WHERE e1.city=e2.city )
```

250

```
SELECT first_name, sales FROM employees e1
WHERE sales >
      (SELECT AVG(sales) FROM employees e2
       WHERE e1.city=e2.city )
```

Correlated subqueries

name	sales	city
Sunita	600	Delhi
Anil	400	Delhi
~~Shanti~~	~~100~~	~~Delhi~~
Sunita	300	Dallas
Maya	300	Dallas
~~Peter~~	~~200~~	~~Dallas~~
~~Max~~	~~100~~	~~Berlin~~
Anna	400	Berlin

```
WHERE sales >
       (SELECT AVG(sales) FROM employees e2
        WHERE e1.city=e2.city )
```

Subquery gets evaluated for every single row!

```
SELECT first_name, sales FROM employees e1
WHERE sales >
     (SELECT AVG(sales) FROM employees e2
      WHERE e1.city=e2.city )
```

Correlated subqueries

name	sales	city
Sunita	600	Delhi
Anil	400	Delhi
[illegible]	[illegible]	[illegible]
[illegible]	[illegible]	[illegible]
[illegible]	[illegible]	[illegible]
[illegible]	[illegible]	[illegible]
Max	100	Berlin
Anna	400	Berlin

```
WHERE sales >
      (SELECT AVG(sales) FROM employees e2
       WHERE e1.city=e2.city )
```

Subquery does not work independently!

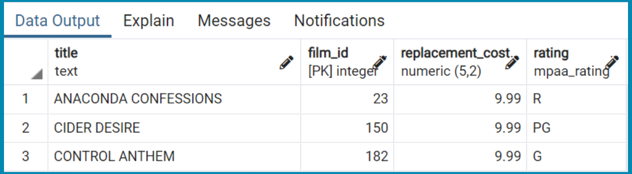
Subquery gets evaluated for every single row!

```
SELECT first_name, sales FROM employees e1
WHERE sales >
     (SELECT AVG(sales) FROM employees e2
      WHERE e1.city=e2.city )
```

Challenge

Show only those movie titles, their associated film_id and replacement_cost with the lowest replacement_costs for in each rating category – also show the rating.

Result

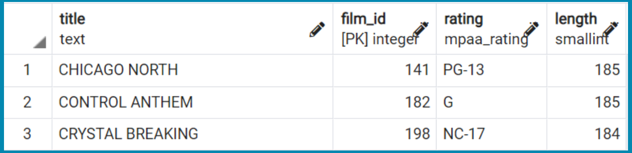


	title text	film_id [PK] integer	replacement_cost numeric (5,2)	rating mpaa_rating
1	ANACONDA CONFESSIONS	23	9.99	R
2	CIDER DESIRE	150	9.99	PG
3	CONTROL ANTHEM	182	9.99	G

Challenge

Show only those movie titles, their associated film_id and the length that have the highest length in each rating category – also show the rating.

Result



	title text	film_id [PK] integer	rating mpaa_rating	length smallint
1	CHICAGO NORTH	141	PG-13	185
2	CONTROL ANTHEM	182	G	185
3	CRYSTAL BREAKING	198	NC-17	184

Correlated subqueries

name	sales	city	min
Sunita	600	Delhi	100
Anil	400	Delhi	100
~~Shanti~~	~~100~~	~~Delhi~~	~~100~~
Sunita	300	Dallas	200
Maya	300	Dallas	200
~~Peter~~	~~200~~	~~Dallas~~	~~200~~
~~Max~~	~~100~~	~~Berlin~~	~~100~~
Anna	400	Berlin	100

```
SELECT first_name, sales FROM employees e1
WHERE sales >
    (SELECT AVG(sales) FROM employees e2
     WHERE e1.city=e2.city )
```

Correlated subqueries

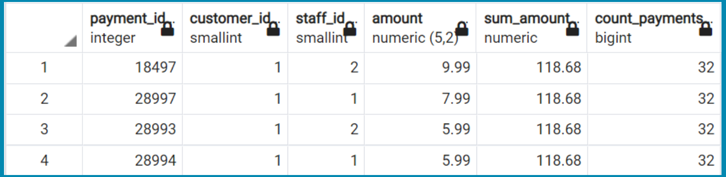
name	sales	city	min
Sunita	600	Delhi	100
Anil	400	Delhi	100
~~Shanti~~	~~100~~	~~Delhi~~	~~100~~
Sunita	300	Dallas	200
Maya	300	Dallas	200
~~Peter~~	~~200~~	~~Dallas~~	~~200~~
~~Max~~	~~100~~	~~Berlin~~	~~100~~
Anna	400	Berlin	100

```
SELECT first_name, sales,
 (SELECT MIN(sales) FROM employees e3
   WHERE e1.city=e3.city )
FROM employees e1
WHERE sales >
     (SELECT AVG(sales) FROM employees e2
      WHERE e1.city=e2.city )
```

Challenge

Show all the payments plus the total amount for every customer as well as the number of payments of each customer.

Result

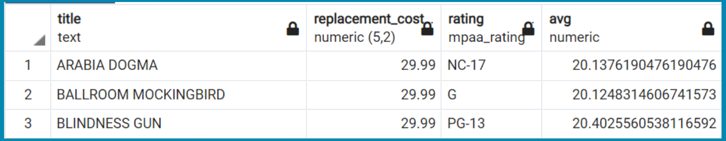


	payment_id integer	customer_id smallint	staff_id smallint	amount numeric (5,2)	sum_amount numeric	count_payments bigint
1	18497	1	2	9.99	118.68	32
2	28997	1	1	7.99	118.68	32
3	28993	1	2	5.99	118.68	32
4	28994	1	1	5.99	118.68	32

Challenge

Show only those films with the highest replacement costs in their rating category plus show the average replacement cost in their rating category.

Result



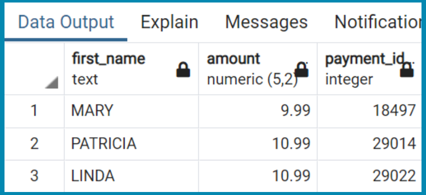
	title text	replacement_cost numeric (5,2)	rating mpaa_rating	avg numeric
1	ARABIA DOGMA	29.99	NC-17	20.1376190476190476
2	BALLROOM MOCKINGBIRD	29.99	G	20.1248314606741573
3	BLINDNESS GUN	29.99	PG-13	20.4025560538116592

Challenge

Show only those payments with the highest payment for each customer's first name - including the payment_id of that payment.

How would you solve it if you would not need to see the payment_id?

Result



	first_name text	amount numeric (5,2)	payment_id integer
1	MARY	9.99	18497
2	PATRICIA	10.99	29014
3	LINDA	10.99	29022