



Relational Algebra

The result of relational operations on relations is also a relation.

$$a, b \in \text{int} \quad \underline{a + b} \in \text{int}$$

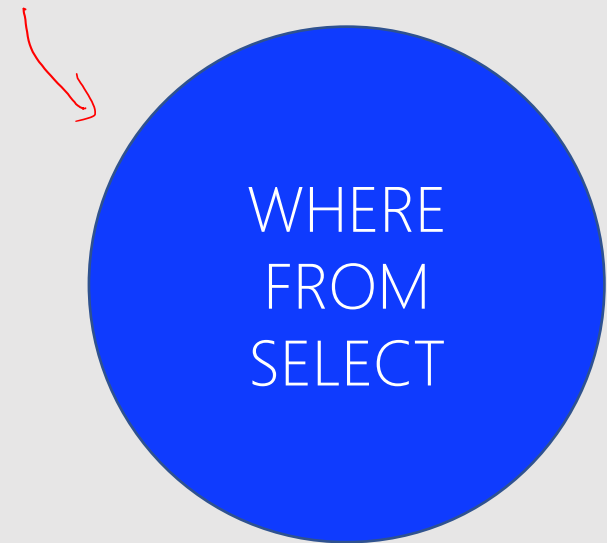
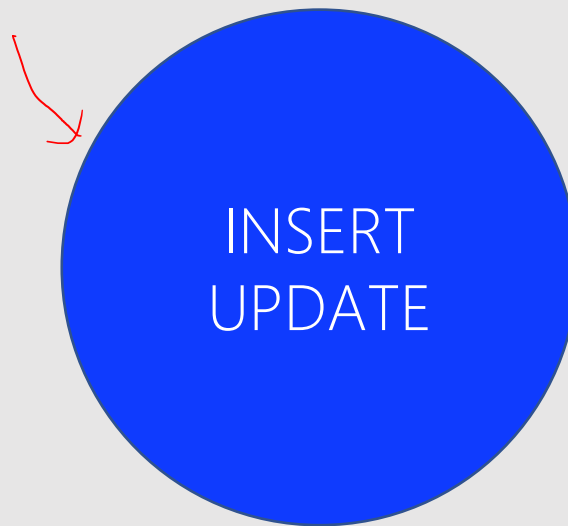
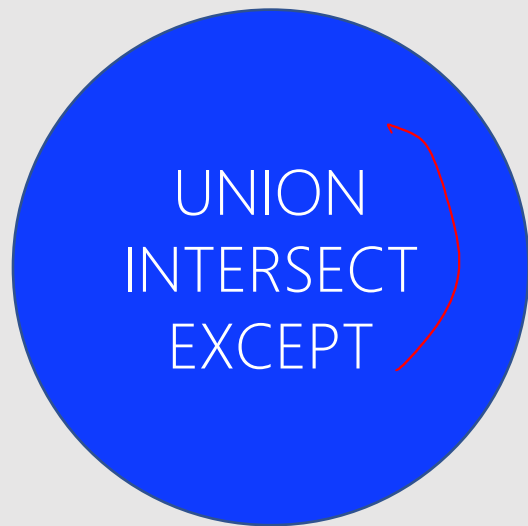
$$a, b \notin \text{int} \quad \underline{a - b}$$

Closure

Advanced SQL × Subquery

The result of SELECT on tables is also a table (temporary though) and can be used inside another query.

Subquery | Nested Query | Inner Query



Advanced SQL × Subquery

The result of **SELECT** on tables is also a table (temporary though) and can be used inside another query.

Subquery | Nested Query | Inner Query



UNION
INTERSECT
EXCEPT

SQL × DML × SELECT

6

1 (SELECT ...)

3 UNION

2 (SELECT ...)

1 (SELECT ...)

3 INTERSECT

2 (SELECT ...)

1 (SELECT ...)

3 EXCEPT

2 (SELECT ...)

Subquery × UNION

FirstName	LastName
Stanley	Kubrick
Clint	Eastwood
John	Travolta
Samuel	Jackson
Uma	Thurman

(SELECT FirstName, LastName FROM Director WHERE PlaceOfBirth='USA')
UNION

(SELECT FirstName, LastName FROM Actor WHERE PlaceOfBirth='USA')

~~SELECT FirstName, LastName FROM Director UNION Actor WHERE PlaceOfBirth='USA'~~ ❌

Subquery × UNION ALL

FirstName	LastName
Stanley	Kubrick
Clint	Eastwood
John	Travolta
Samuel	Jackson
Uma	Thurman
Clint	Eastwood

(SELECT FirstName, LastName FROM Director WHERE PlaceOfBirth='USA')
UNION ALL
(SELECT FirstName, LastName FROM Actor WHERE PlaceOfBirth='USA')

Subquery × INTERSECT

FirstName	LastName
Clint	Eastwood

(SELECT FirstName, LastName FROM Director)

INTERSECT

(SELECT FirstName, LastName FROM Actor)

~~SELECT FirstName, LastName FROM Director INTERSECT Actor~~ ❌

Subquery × EXCEPT

FirstName	LastName
Stanley	Kubrick
Alfred	Hitchcock

(SELECT FirstName, LastName FROM Director)
EXCEPT

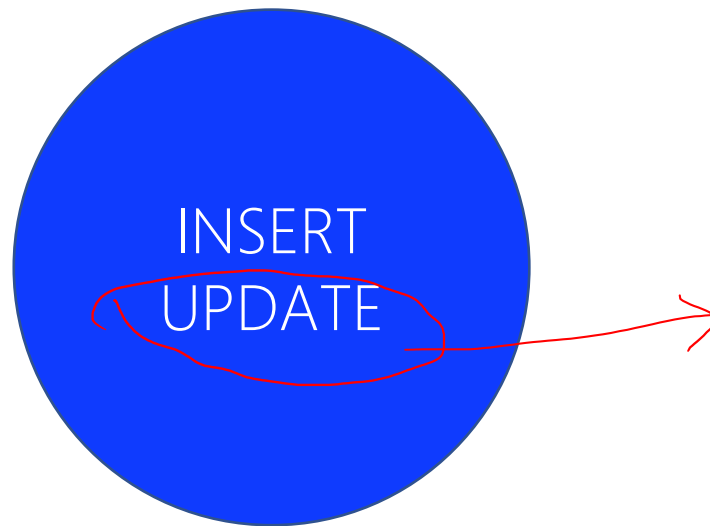
(SELECT FirstName, LastName FROM Actor)

~~SELECT~~ FirstName, LastName ~~FROM~~ Director ~~EXCEPT~~ Actor ❌

Advanced SQL × Subquery

The result of **SELECT** on tables is also a table (temporary though) and can be used inside another query.


Subquery | Nested Query | Inner Query



Subquery × Bulk × INSERT

Subquery can be used to do bulk INSERT.

INSERT INTO TableName(c, c', c'', ...) (SELECT ...);



Subquery × Bulk × INSERT

Subquery can be used to do bulk INSERT.

INSERT INTO TableName(^{Int}c, c', c'', ...) (SELECT ...);



- The number of columns must be same.
- The data type of columns must be compatible.

Subquery × Bulk × INSERT

Subquery can be used to do bulk INSERT.

INSERT INTO *TableName*(*c*, *c'*, *c''*, ...) (SELECT ...);

- The number of columns must be same.
- The data type of columns must be compatible.

INSERT INTO *TableName* (SELECT ...);



- If column list is omitted, the order in *TableName's definition* is assumed.

Subquery × Bulk × INSERT

Subquery can be used to do bulk INSERT.

INSERT INTO *TableName*(*c*, *c'*, *c''*, ...) (SELECT ...);

- The number of columns must be same.
- The data type of columns must be compatible.

INSERT INTO *TableName* (SELECT ...);



- If column list is omitted, the order in *TableName's definition* is assumed.

In SQLite, drop the parenthesis around subquery.

Subquery × Bulk × INSERT

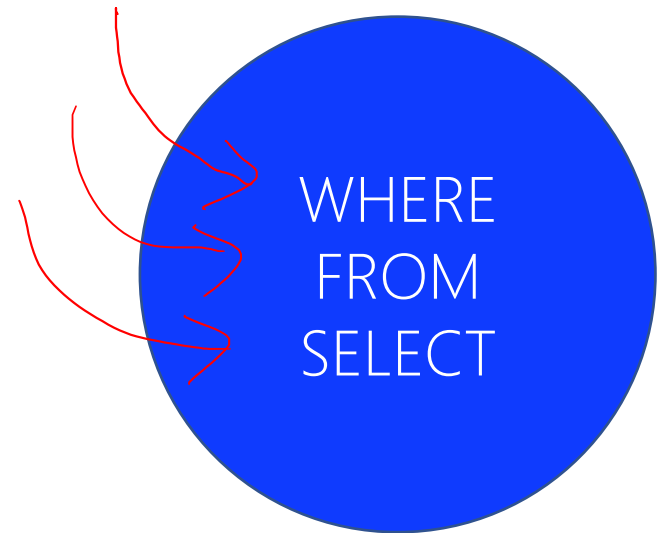
16

INSERT INTO OldMovie (C_1, C_2) \rightarrow $\text{intu into } T(C_1, C_2)$
(SELECT * FROM Movie WHERE ReleaseDate < 1940) $(\text{select}(\text{self}_{C_1}), \text{self}_{C_2})$

Advanced SQL × Subquery

The result of **SELECT** on tables is also a table (temporary though) and can be used inside another query.

Subquery | Nested Query | Inner Query



Subquery × WHERE

Subquery can be used in WHERE clause of DML statements, i.e., SELECT, UPDATE, INSERT INTO, & DELETE FROM.

Here, we use WHERE clause in SELECT statement as it is the most common DML.



Subquery × WHERE

4 SELECT Columns
1 FROM Tables
3 WHERE (c₁, c₂, ..., c_n) OP (2 SELECT c'₁, c'₂, ..., c'_n FROM ...)

- OP ∈ {=, ≥, >=, <, <=, <>}
- Subquery cannot have more than one row, BUT can have multiple columns
- Subquery always on right side of OP.

Subquery × WHERE × ~~AGG~~

Movie				
Id	Title	Language	ReleaseDate	RunningTime
1	2001: A Space Odyssey	English	1968	142
2	Rosemary's Baby	English	1968	NULL
3	The Birds	English	1963	119
4	Planet of the Apes	EN	1968	112

What is the oldest movie?

```
SELECT *  
FROM Movie  
WHERE ReleaseDate = MIN(ReleaseDate)
```



Subquery × WHERE × AGG

Movie				
Id	Title	Language	ReleaseDate	RunningTime
1	2001: A Space Odyssey	English	1968 =	142
2	Rosemary's Baby	English	1968	NULL
3	The Birds	English	1963	119
4	Planet of the Apes	EN	1968	112

What is the oldest movie?

SELECT *
 FROM Movie
 WHERE ReleaseDate = (SELECT MIN(ReleaseDate) FROM Movie)

100
 101 63
 112
 115
 2022
 2019

group By
 real

Subquery × WHERE × ~~AGG~~

Movie				
Id	Title	Language	ReleaseDate	RunningTime
1	2001: A Space Odyssey	English	1968	142
2	Rosemary's Baby	English	1968	NULL
3	The Birds	English	1963	119
4	Planet of the Apes	EN	1968	112

What are the longest movies in 1963?

```

SELECT *
FROM Movie
WHERE ReleaseDate = 1963 AND
RunningTime = MAX(RunningTime)

```



Subquery × WHERE × AGG

Movie				
Id	Title	Language	ReleaseDate	RunningTime
1	2001: A Space Odyssey	English	1968	142
2	Rosemary's Baby	English	1968	NULL
3	The Birds	English	1963	119
4	Planet of the Apes	EN	1968	112

What are the longest movies in 1963?

SELECT *
 FROM Movie
 WHERE ReleaseDate = 1963 AND
 RunningTime = (SELECT MAX(RunningTime) FROM Movie)

142

✗

Subquery × WHERE × AGG

Movie				
Id	Title	Language	ReleaseDate	RunningTime
1	2001: A Space Odyssey	English	1968	142
2	Rosemary's Baby	English	1968	NULL
3	The Birds	English	1963	119 ✓
4	Planet of the Apes	EN	1968	112

What are the longest movies in 1963?

```

SELECT *
FROM Movie
WHERE ReleaseDate = 1963
      RunningTime = (SELECT MAX(RunningTime)
                     FROM Movie
                     WHERE ReleaseDate = 1963)
  
```

Handwritten annotations: "119" above the subquery, a red circle around "MAX", and a red checkmark next to the subquery result.

Subquery × WHERE × AGG

Movie				
Id	Title	Language	ReleaseDate	RunningTime
1	2001: A Space Odyssey	English	1968	142
2	Rosemary's Baby	English	1968	NULL
3	The Birds	English	1963	119
4	Planet of the Apes	EN	1968	112

What is the oldest, yet longest movie?

```

SELECT *
FROM Movie
WHERE ReleaseDate = (SELECT MIN(ReleaseDate) FROM Movie) AND
      RunningTime = (SELECT MAX(RunningTime) FROM Movie)
  
```

Handwritten annotations: Red circles around 1, 2, and 3 in the SQL query. Red arrows point from the circles to the corresponding values in the table: 1 points to 142, 2 points to 1963, and 3 points to 112. A red circle with a cross is also present.

Subquery × WHERE × AGG

Movie				
Id	Title	Language	ReleaseDate	RunningTime
1	2001: A Space Odyssey	English	1968	142
2	Rosemary's Baby	English	1968	NULL
3	The Birds	English	1963	119
4	Planet of the Apes	EN	1968	112

What is the oldest, yet longest movie?

```

SELECT *
FROM Movie
WHERE (ReleaseDate, RunningTime) = (SELECT MIN(ReleaseDate), MAX(RunningTime)
                                     FROM Movie)

```

1963, 142

Subquery × WHERE × AGG

Movie				
Id	Title	Language	ReleaseDate	RunningTime
1	2001: A Space Odyssey	English	1968	142
2	Rosemary's Baby	English	1968	NULL
3	The Birds	English	1963	119
4	Planet of the Apes	EN	1968	112

What movies are shorter than average length of movies in 1968?

```

SELECT *
FROM Movie
WHERE RunningTime < (SELECT AVG(RunningTime)
                      FROM Movie
                      WHERE ReleaseDate = 1968)

```

$$\frac{142 + 112}{2} = 127$$

Subquery × WHERE × AGG

28

Movie				
Id	Title	Language	ReleaseDate	RunningTime
1	2001: A Space Odyssey	English	1968	142
2	Rosemary's Baby	English	1968	NULL
3	The Birds	English	1963	119
4	Planet of the Apes	EN	1968	112

What movies in 1968 are shorter than average length of movies?

Subquery × WHERE × AGG

29

Movie				
Id	Title	Language	ReleaseDate	RunningTime
1	2001: A Space Odyssey	English	1968	142
2	Rosemary's Baby	English	1968	NULL
3	The Birds	English	1963	119
4	Planet of the Apes	EN	1968	112

average length of movies?

(SELECT AVG(RunningTime)
FROM Movie)

$\frac{142 + 119 + 112}{3}$

Subquery × WHERE × AGG

Movie				
Id	Title	Language	ReleaseDate	RunningTime
1	2001: A Space Odyssey	English	1968	142
2	Rosemary's Baby	English	1968	NULL
3	The Birds	English	1963	119
4	Planet of the Apes	EN	1968	112

What movies in 1968 are shorter than average length of movies?

SELECT *

FROM Movie

WHERE ReleaseDate = 1968

AND

RunningTime < (SELECT AVG(RunningTime)
FROM Movie)

3

142 + 119 + 112

112

<

WHERE × IN

31

Movie				
Id	Title	Language	ReleaseDate	RunningTime
1	2001: A Space Odyssey	English	1968	142
2	Rosemary's Baby	English	1968	NULL
3	The Birds	English	1963	119
4	Planet of the Apes	EN	1968	112

Find the movies in English language?

SELECT *

FROM Movie

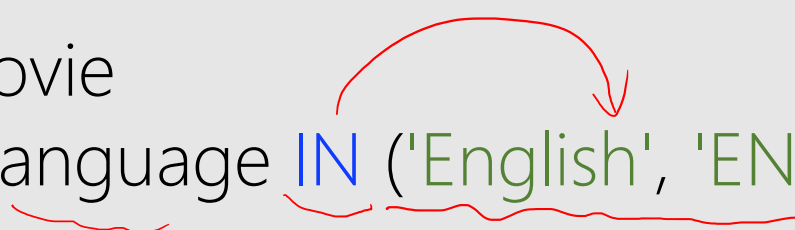
WHERE Language = 'English' OR
Language = 'EN'

WHERE × IN

Movie				
Id	Title	Language	ReleaseDate	RunningTime
1	2001: A Space Odyssey	English	1968	142
2	Rosemary's Baby	English	1968	NULL
3	The Birds	English	1963	119
4	Planet of the Apes	EN	1968	112

Find the movies in English language?

```
SELECT *  
FROM Movie  
WHERE Language IN ('English', 'EN')
```



WHERE × IN

Movie				
Id	Title	Language	ReleaseDate	RunningTime
1	2001: A Space Odyssey	English	1968	142
2	Rosemary's Baby	English	1968	NULL
3	The Birds	English	1963	119
4	Planet of the Apes	EN	1968	112

Find the movies in non-English language?

```
SELECT *  
FROM Movie  
WHERE NOT (Language = 'English' OR  
             Language = 'EN')
```

(A or B)

WHERE × IN

Movie				
Id	Title	Language	ReleaseDate	RunningTime
1	2001: A Space Odyssey	English	1968	142
2	Rosemary's Baby	English	1968	NULL
3	The Birds	English	1963	119
4	Planet of the Apes	EN	1968	112

Find the movies in non-English language?

```
SELECT *
FROM Movie
WHERE Language <> 'English' AND
       Language <> 'EN'
```

$$\overline{(A + B)} = \bar{A} \text{ AND } \bar{B}$$

*De Morgan's law: NOT (A OR B) = NOT(A) AND NOT(B)

WHERE × IN

Movie				
Id	Title	Language	ReleaseDate	RunningTime
1	2001: A Space Odyssey	English	1968	142
2	Rosemary's Baby	English	1968	NULL
3	The Birds	English	1963	119
4	Planet of the Apes	EN	1968	112

Find the movies in non-English language?

```
SELECT *
FROM Movie
WHERE Language NOT IN ('English', 'EN')
```

$\overline{A \text{ OR } B} = \overline{A} \text{ AND } \overline{B}$

$\overline{(A + B)} = \overline{A \text{ OR } B}$

Subquery × WHERE × IN

4 SELECT Columns
1 FROM Tables
3 WHERE (c₁, c₂, ..., c_n) [NOT] IN (SELECT c'₁, c'₂, ..., c'_n FROM ...)

The diagram illustrates the syntax of a SQL WHERE clause using the IN operator. It shows the sequence of keywords: SELECT, FROM, WHERE, and IN. The WHERE clause is annotated with a red circle and a blue circle containing the number 2. The IN operator is circled in red. The subquery (SELECT c'₁, c'₂, ..., c'_n FROM ...) is enclosed in a red oval. A red arrow points from the subquery back to the WHERE clause, indicating the relationship between the two. The columns c₁, c₂, ..., c_n in the WHERE clause and c'₁, c'₂, ..., c'_n in the subquery are highlighted in yellow.

- Subquery can have multiple rows and columns.
- Subquery always on right side of IN.

Subquery × WHERE × IN

Director						
Id	FirstName	LastName	DateOfBirth	PlaceOfBirth	BestMovieId	MovieCount
1	Stanley	Kubrick	Jul. 26, 1928	USA	1	13
2	Alfred	Hitchcock	Aug. 13, 1899	England	NULL	47
3	Clint	Eastwood	May 31, 1930	USA	NULL	35

Movie			
Id	Title	Language	RunningTime
1	2001: A Space Odyssey	English	142
2	Rosemary's Baby	English	NULL

What movies are directors' best movies?

```

SELECT *
FROM Movie
WHERE Id IN (SELECT BestMovieId FROM Director)

```

Handwritten notes: "2 IN" above the WHERE clause, and "N" below the Id column.

Handwritten notes: A bracket on the right side of the query, with "1 ✓" above it and "NULL" and "NULL" below it.

Subquery × WHERE × IN

Director						
Id	FirstName	LastName	DateOfBirth	PlaceOfBirth	BestMovieId	MovieCount
1	Stanley	Kubrick	Jul. 26, 1928	USA	1	13
2	Alfred	Hitchcock	Aug. 13, 1899	England	NULL	47
3	Clint	Eastwood	May 31, 1930	USA	NULL	35

Movie			
Id	Title	Language	RunningTime
1	2001: A Space Odyssey	English	142
2	Rosemary's Baby	English	NULL

What movies are directors' best movies?

SELECT *
 FROM Movie
 WHERE Id = (SELECT BestMovieId FROM Director)

1 = (NULL
NULL)



FROM × INNER JOIN

Director						
Id	FirstName	LastName	DateOfBirth	PlaceOfBirth	BestMovieId	MovieCount
1	Stanley	Kubrick	Jul. 26, 1928	USA	1	13
2	Alfred	Hitchcock	Aug. 13, 1899	England	NULL	47
3	Clint	Eastwood	May 31, 1930	USA	NULL	35

Movie			
Id	Title	Language	RunningTime
1	2001: A Space Odyssey	English	142
2	Rosemary's Baby	English	NULL

What movies are directors' best movies?

```

SELECT M.* *
FROM Movie AS M
INNER JOIN Director AS D ON M.Id = D.BestMovieId

```

WHERE × IN vs. FROM × INNER JOIN

What movies are directors' best movies?

```

SELECT *
FROM Movie
WHERE Id IN (SELECT BestMovieId FROM Director)

```

Handwritten annotations:
 - Under Movie: 1k
 - Under Id: 1k
 - Under BestMovieId: 2k
 - Above FROM Director: 1k * 2k
 - Below SELECT BestMovieId FROM Director: { 2k

Which one?

```

SELECT M.*
FROM Movie AS M
INNER JOIN Director AS D ON M.Id = D.BestMovieId


```

Handwritten annotations:
 - Under M.*: 1k
 - Under Movie AS M: 1k
 - Under Director AS D: 2k
 - Under ON M.Id = D.BestMovieId: 1k * 2k
 - To the right of the join condition: } x
 - Above the join condition: → 500

WHERE × IN vs. FROM × INNER JOIN

What movies are directors' best movies?

```
SELECT *  
FROM Movie  
WHERE Id IN (SELECT BestMovieId FROM Director)
```



Which one? We can access Director info in below query.

```
SELECT M.*, D.*  
FROM Movie AS M  
INNER JOIN Director AS D ON M.Id = D.BestMovieId
```



WHERE × IN vs. FROM × INNER JOIN

What movies are directors' best movies?

```
SELECT *  
FROM Movie  
WHERE Id IN (SELECT BestMovieId FROM Director)
```

Which one? I don't need Director info! Which one is faster?

```
SELECT M.*  
FROM Movie AS M  
INNER JOIN Director AS D ON M.Id = D.BestMovieId
```



WHERE × IN vs. INTERSECT

FirstName	LastName
Clint	Eastwood

(SELECT FirstName, LastName FROM Director)

INTERSECT

(SELECT FirstName, LastName FROM Actor)

SELECT *

FROM Director

WHERE (FirstName, LastName) IN (SELECT FirstName, LastName FROM Actor)

MySQL does not support INTERSECT.

WHERE × IN vs. EXCEPT

FirstName	LastName
Stanley	Kubrick
Alfred	Hitchcock

{ (SELECT FirstName, LastName FROM Director)
EXCEPT
(SELECT FirstName, LastName FROM Actor)

{ SELECT *
FROM Director
WHERE (FirstName, LastName) NOT IN (SELECT FirstName, LastName FROM Actor)

MySQL does not support EXCEPT.