Extra fundamentals for table

Reja:

1. Subquery, ANY, ALL, EXISTS
2. Set operations (UNION, Intersect, Except)
3. CONDITIONAL EXPRESSIONS & OPERATORS (case, coalesce, nullif, cast)

Subquery

Subquery - Quyi so`rov yoki Ichki so`rov yoki Nested so`rov boshqa PostgreSQL so`rovi ichidagi so`rovdir va WHERE bandiga kiritilgan.

Quyi so`rov asosiy so`rovda olinadigan ma`lumotlarni yanada cheklash sharti sifatida foydalaniladigan ma`lumotlarni qaytarish uchun ishlatiladi.

Quyi so`rovlar SELECT, INSERT, UPDATE va DELETE operatorlari bilan birga =, <, >, >=, <=, IN va hokazo operatorlar bilan ishlatilishi mumkin.

Quyi so`rovlar amal qilishi kerak bo`lgan bir nechta qoidalar mavjud -

- Quyi so`rovlar qavs ichiga olinishi kerak.
- Quyi so`rov tanlangan ustunlarini solishtirish uchun asosiy so`rovda bir nechta ustunlar bo`lmasa, SELECT bandida faqat bitta ustun bo`lishi mumkin.
- ORDER BY ni quyi so`rovda ishlatib bo`lmaydi, lekin asosiy so`rov ORDER BY dan foydalanishi mumkin. GROUP BY quyi so`rovdagi ORDER BY funksiyasini bajarish uchun ishlatilishi mumkin.
- Bir nechta qator qaytaradigan quyi so`rovlar faqat bir nechta qiymat operatorlari bilan ishlatilishi mumkin, masalan, IN, EXISTS, NOT IN, ANY/SOME, ALL operatorlari.
- BETWEEN operatoridan quyi so`rov bilan foydalanib bo`lmaydi, ammo BETWEEN pastki so`rov ichida ishlatilishi mumkin.

Subquery

Subquery Syntax:

```
Select    select_list
From      table
Where     expr operator
                ( Select    select_list
                  From      table );
```

Subquery

```
SELECT first_name,last_name, salary
FROM employees
WHERE salary >
(SELECT max(salary) FROM employees
WHERE first_name='Alexander');
```

first_name	last_name	salary
Steven	King	24000.00
Neena	Kochhar	17000.00
Lex	De Haan	17000.00
Nancy	Greenberg	12000.00
Den	Raphaely	11000.00
John	Russell	14000.00
Karen	Partners	13500.00
Alberto	Errazuriz	12000.00
Gerald	Cambrault	11000.00
Eleni	Zlotkey	10500.00
Peter	Tucker	10000.00
David	Bernstein	9500.00
Janette	King	10000.00

Subquery

```
SELECT
        film_id,
        title,
        rental_rate
FROM
        film
WHERE
        rental_rate > (
                SELECT
                        AVG (rental_rate)
                FROM
                        film);
```

film_id	title	rental_rate
133	Chamber Italian	4.99
384	Grosse Wonderful	4.99
8	Airport Pollock	4.99
98	Bright Encounters	4.99
2	Ace Goldfinger	4.99
3	Adaptation Holes	2.99
4	Affair Prejudice	2.99
5	African Egg	2.99

ANY

ANY operatori qiymatni quyi so`rov tomonidan qaytarilgan har bir qiymatga solishtiradi. Shuning uchun ANY kalit so`z agar quyi so`rov qaytaradigan ustundagi har qanday qiymat uchun taqqoslash TRUE bo`lsa, TRUE qaytaradi.

```
SELECT first_name, last_name,department_id
FROM employees
WHERE department_id= ANY
(SELECT DEPARTMENT_ID
FROM departments WHERE location_id=1700);
```

first_name	last_name	department_id
Steven	King	90
Neena	Kochhar	90
Lex	De Haan	90
Nancy	Greenberg	100
Daniel	Faviet	100
John	Chen	100
Ismael	Sciarra	100
Jose Manuel	Urman	100
Luis	Popp	100
Den	Raphaely	30
Alexander	Khoo	30
Shelli	Baida	30
Sigal	Tobias	30
Guy	Himuro	30
Karen	Colmenares	30
Jennifer	Whalen	10
Shelley	Higgins	110
William	Gietz	110

(18 rows)

ALL

PostgreSQL **ALL** operatori qiymatni pastki so`rov tomonidan qaytarilgan qiymatlar ro`yxati bilan solishtirish orqali ma`lumotlarni so`rash imkonini beradi .

•ALL operatoridan oldin teng (=), teng emas (!=), katta (>), katta yoki teng (>=), kichik (<) va kichik yoki yoki kabi taqqoslash operatori bo‘lishi kerak. (<=) ga teng.
•ALL operatoridan keyin pastki so'rov bo'lishi kerak, u ham qavslar bilan o'ralgan bo'lishi kerak.
Quyi so'rov ba'zi qatorlarni qaytaradi degan faraz bilan ALL operatori quyidagicha ishlaydi:
1.column_name > ALL (subquery) Agar qiymat quyi so'rov tomonidan qaytarilgan eng katta qiymatdan katta bo'lsa, ifoda rost deb baholanadi.
2.column_name >= ALL (subquery) Agar qiymat quyi so'rov tomonidan qaytarilgan eng katta qiymatdan katta yoki unga teng bo'lsa, ifoda rost deb baholanadi.
3.column_name < ALL (subquery) Agar qiymat pastki so'rov tomonidan qaytarilgan eng kichik qiymatdan kichik bo'lsa, ifoda rost deb baholanadi.
4.column_name <= ALL (subquery) Agar qiymat pastki so'rov tomonidan qaytarilgan eng kichik qiymatdan kichik yoki unga teng bo'lsa, ifoda rost deb baholanadi.
5.column_name = ALL (subquery) Agar qiymat quyi so'rov tomonidan qaytarilgan har qanday qiymatga teng bo'lsa, ifoda rost deb baholanadi.
6.column_name != ALL (subquery) Agar qiymat pastki so'rov tomonidan qaytarilgan qiymatga teng bo'lmasa, ifoda rost deb baholanadi.

ALL

```
SELECT film_id,title,length
FROM film
WHERE length > ALL (SELECT ROUND(AVG (length),2)
        FROM film
        GROUP BY rating)
ORDER BY length;
```

film_id	title	length
37	Arizona Bang	121
86	Boogie Amelie	121
93	Brannigan Sunrise	121
207	Dangerous Uptown	121
403	Harry Idaho	121
490	Jumanji Blade	121
658	Paris Weekend	121
704	Pure Runner	121
58	Beach Heartbreakers	122
68	Betrayed Rear	122
142	Chicken Hellfighters	122
175	Confused Candles	122
218	Deceiver Betrayed	122

EXISTS

EXISTS quyi so`rov (subquery) har qanday satrlarni qaytaradimi yoki yo`qligini aniqlash uchun ishlatiladi. Agar quyi so`rov  kamida bitta qatorni qaytarsa, EXISTS natijasi TRUE, agar quyi so`rov hech qanday qatorni qaytarmasa, EXISTS natijasi FALSE.

Properties SQL Statistics Dependencies Dependents javatpoint/postgres@PostgreSQL 12 *

No limit

javatpoint/postgres@PostgreSQL 12

Query Editor Query History

```
1  SELECT *
2  FROM Client
3  WHERE EXISTS (SELECT 1
4                FROM Client_details
5                WHERE Client.client_id = client_details.client_id);
```

Data Output Messages Explain Notifications

	client_id [PK] integer	client_name character varying	client_profession character varying	client_qualification character varying	client_salary integer
1	1	Emma Hernandez	Web Designer	BTech	25000
2	2	Mia Clark	Software Engineer	BE	20000
3	3	Noah Rodriguez	Bussinessman	MBA	50000
4	4	Martha Brown	Doctor	MBBS	75000
5	5	James Luther	HR	MBA	35000

Set operations

Postgres ma`lumotlar bazasi qidiruv natijalarini so`rash va filtrlashni osonlashtiradigan operatorlarni taklif qiladi. Set operatorlari ikki yoki undan ortiq SELECT iboralari natijalarini birlashtirish uchun ishlatiladi. Bu operatorlar **UNION**, **UNION ALL**, **INTERSECT** va **EXCEPT** - har biridan bir nechta jadvallar bo`ylab so`rovlar tuzish va kerakli ma`lumotlarni filtrlash uchun foydalanish mumkin.

UNION

PostgreSQL **UNION** operatori ikki yoki undan ortiq SELECT iboralari natijalarini takroriy qatorlarni qaytarmasdan birlashtirish uchun ishlatiladi.
UNION dan foydalanish uchun har bir SELECTda bir xil miqdordagi ustunlar, bir xil miqdordagi ustun ifodalari, bir xil ma`lumotlar turi tanlangan bo`lishi va ular bir xil tartibda bo`lishi kerak, lekin ularning uzunligi bir xil bo`lishi shart emas.

```
SELECT * FROM top_rated_films
UNION
SELECT * FROM most_popular_films;
```

	title character varying	release_year smallint
1	An American Pickle	2020
2	Greyhound	2020
3	The Shawshank Redemption	1994
4	The Godfather	1972
5	12 Angry Men	1957

UNION ALL

UNION ALL operatori ikkita SELECT bayonoti natijalarini, shu jumladan takroriy qatorlarni birlashtirish uchun ishlatiladi. UNION uchun amal qiladigan qoidalar UNION ALL operatoriga ham tegishli.

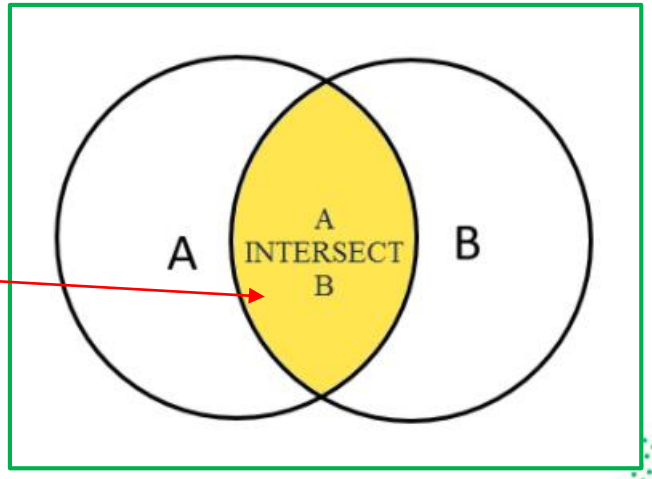
```
SELECT * FROM top_rated_films
UNION ALL
SELECT * FROM most_popular_films;
```

	title character varying	release_year smallint
1	The Shawshank Redemption	1994
2	The Godfather	1972
3	12 Angry Men	1957
4	An American Pickle	2020
5	The Godfather	1972
6	Greyhound	2020

INTERSECT

PostgreSQLda tanlangan ma`lumotlarning umumiy qatorlarini dublikatsiz va saralangan tartibda joylashtirilgan ma`lumotlar bilan ko`rsatish uchun INTERSECT operatoridan foydalaniladi.

```
SELECT select_list
FROM A
INTERSECT
SELECT select_list
FROM B;
```



INTERSECT

The `top_rated_films` table:

	title character varying	release_year smallint
1	The Shawshank Redemption	1994
2	The Godfather	1972
3	12 Angry Men	1957

The `most_popular_films` table:

	title character varying	release_year smallint
1	An American Pickle	2020
2	The Godfather	1972
3	Greyhound	2020

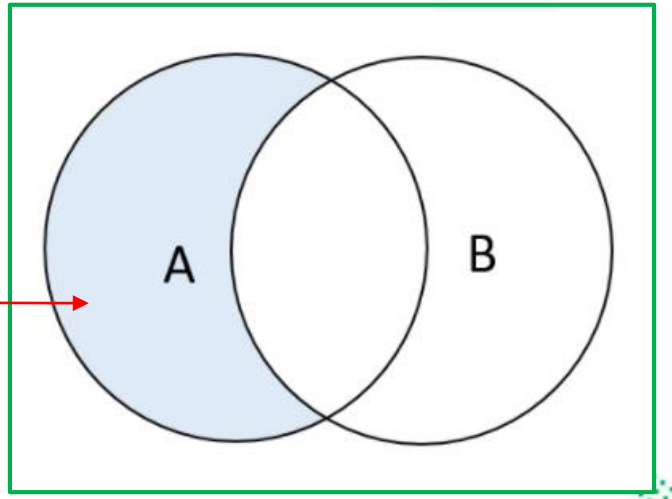
```
SELECT *
FROM most_popular_films
INTERSECT
SELECT *
FROM top_rated_films;
```

	title character varying	release_year smallint
1	The Godfather	1972

EXCEPT

Birinchi jadvalda mavjud bo`lgan, ikkinchisida yo`q bo`lgan ma`lumotlarni ko`rsatish uchun EXCEPT operatoridan foydalaniladi. U hech qanday dublikatni qaytarmaydi va ma`lumotlar default holatda o`sish tartibida tartibga solinadi.

```
SELECT select_list
FROM A
EXCEPT
SELECT select_list
FROM B;
```



EXCEPT

The `top_rated_films` table:

	title character varying	release_year smallint
1	The Shawshank Redemption	1994
2	The Godfather	1972
3	12 Angry Men	1957

The `most_popular_films` table:

	title character varying	release_year smallint
1	An American Pickle	2020
2	The Godfather	1972
3	Greyhound	2020

```
SELECT * FROM top_rated_films
EXCEPT
SELECT * FROM most_popular_films;
```

	title character varying	release_year smallint
1	The Shawshank Redemption	1994
2	12 Angry Men	1957

CONDITIONAL EXPRESSIONS & OPERATORS

- CASE
- COALESCE
- NULLIF
- CAST

CASE

PostgreSQLda CASE ifodasi boshqa dasturlash tillaridagi IF/ELSE bilan bir xil. Bu kuchli so`rovni shakllantirish uchun so`rovga if-else mantiqini qo`shish imkonini beradi.
CASE ni SELECT, WHERE, GROUP BY, va HAVING bilan birga foydalanish mumkin.
CASE ning ikki xil shakllari bor: **Umumiy va oddiy shakli**

```
CASE
    WHEN condition_1  THEN result_1
    WHEN condition_2  THEN result_2
    [WHEN ...]
    [ELSE else_result]
END
```

```
CASE expression
  WHEN value_1 THEN result_1
  WHEN value_2 THEN result_2
  [WHEN ...]
ELSE
  else_result
END
```

UMUMIY CASE

```
SELECT title,
       length,
       CASE
           WHEN length> 0
                AND length <= 50 THEN 'Short'
           WHEN length > 50
                AND length <= 120 THEN 'Medium'
           WHEN length> 120 THEN 'Long'
       END duration
FROM film
ORDER BY title;
```

	title character varying (255)	length smallint	duration text
1	Academy Dinosaur	86	Medium
2	Ace Goldfinger	48	Short
3	Adaptation Holes	50	Short
4	Affair Prejudice	117	Medium
5	African Egg	130	Long
6	Agent Truman	169	Long
7	Airplane Sierra	62	Medium
8	Airport Pollock	54	Medium
9	Alabama Devil	114	Medium
10	Aladdin Calendar	63	Medium
11	Alamo Videotape	126	Long
12	Alaska Phantom	136	Long
13	Ali Forever	150	Long
14	Alice Fantasia	94	Medium

ODDIY CASE

```sql
SELECT title,
       rating,
       CASE rating
           WHEN 'G' THEN 'General Audiences'
           WHEN 'PG' THEN 'Parental Guidance Suggested'
           WHEN 'PG-13' THEN 'Parents Strongly Cautioned'
           WHEN 'R' THEN 'Restricted'
           WHEN 'NC-17' THEN 'Adults Only'
       END rating_description
FROM film
ORDER BY title;
```

	title character varying (255)	rating mpaa_rating	rating_description text
1	Academy Dinosaur	PG	Parental Guidance Suggested
2	Ace Goldfinger	G	General Audiences
3	Adaptation Holes	NC-17	Adults Only
4	Affair Prejudice	G	General Audiences
5	African Egg	G	General Audiences
6	Agent Truman	PG	Parental Guidance Suggested
7	Airplane Sierra	PG-13	Parents Strongly Cautioned
8	Airport Pollock	R	Restricted
9	Alabama Devil	PG-13	Parents Strongly Cautioned
10	Aladdin Calendar	NC-17	Adults Only
11	Alamo Videotape	G	General Audiences
12	Alaska Phantom	PG	Parental Guidance Suggested

COALESCE

PostgreSQLda COALESCE funksiyasi birinchi null bo`lmagan argumentni qaytaradi. Funksiya argumentlarni cheksiz ko`p qabul qiladi. U null bo`lmagan birinchi argumentni qaytaradi. Agar barcha argumentlar null bo`lsa, COALESCE funksiya nullni qaytaradi.

```
COALESCE (argument_1, argument_2, …);
```

COALESCE

```
SELECT
        COALESCE (1, 2);
```

coalesce
1

```
SELECT
        COALESCE (NULL, 2 , 1);
```

coalesce
2

NULLIF

NULLIF - bu PostgreSQLda null qiymatlar yoki ifodalarni boshqarish uchun ishlatiladigan keng tarqalgan shartli ifoda. NULLIF, shuningdek, null qiymatlarni boshqarish uchun COALESCE funksiyasi bilan birga ishlatiladi. PostgreSQL NULLIF funksiyasi, agar taqdim etilgan ifodalar teng bo`lsa, null qiymatini qaytaradi, aks holda, natijada birinchi ifodani qaytaradi.

```
NULLIF(argument_1,argument_2);
```

NULLIF

```
SELECT
        NULLIF (1, 1); -- return NULL

SELECT
        NULLIF (1, 0); -- return 1

SELECT
        NULLIF ('A', 'B'); -- return A
```

CAST

Bir nechta ma`lumotlar bazalarida tranzaksiyalarni amalga oshirishda ma`lumotlarni konvertatsiya qilish deyarli barcha dasturlash tillari tomonidan qo`llab-quvvatlanadigan asosiy talabdir. PostgreSQL bizga CAST operatorini taqdim etadi, biz undan bitta ma`lumot turini boshqa ma`lumot turiga aylantirish uchun foydalanishimiz mumkin. PostgreSQLda biz turli xil CAST operatsiyalariga ega bo`lishimiz mumkin, masalan, satrni butun sonlarga o`tkazish, satrni sana va sanani satrga o`tkazish va hokazo.

Quyida CAST turidagi sintaksik tasvirlangan:

```
CAST ( expression AS target_type );
```

CAST sintaksisidan tashqari, bir turdagi qiymatni boshqasiga aylantirish uchun quyidagi sintaksisdan foydalanishingiz mumkin:

```
expression::type
```

CAST

```
SELECT
        CAST ('100' AS INTEGER);
```

```
SELECT
  '100'::INTEGER,
  '01-OCT-2015'::DATE;
```

Agar ma`lumot maqsadli turga aylantirilmasa, PostgreSQL xatoga yo`l qo`yadi. Quyidagi misolga qarang:

```
SELECT
        CAST ('10C' AS INTEGER);
```

```
[Err] ERROR:  invalid input syntax for integer: "10C"
LINE 2:  CAST ('10C' AS INTEGER);
```

```
SELECT
        CAST ('10.2' AS DOUBLE);
```

Voy, biz quyidagi xato xabarini oldik:

```
[Err] ERROR:  type "double" does not exist
LINE 2:  CAST ('10.2' AS DOUBLE)
```

Buni tuzatish uchun siz quyidagi `DOUBLE PRECISION` o'rniga foydalanishingiz kerak `DOUBLE`:

```
SELECT
   CAST ('10.2' AS DOUBLE PRECISION);
```

	float8 double precision
1	10.2

PDP ACADEMY

E`TIBORINGIZ UCHUN RAHMAT!