

NOTES 4 SELECT WHERE

The WHERE syntax allows us to do comparisons: Where are column values greater than or less than 10, etc.

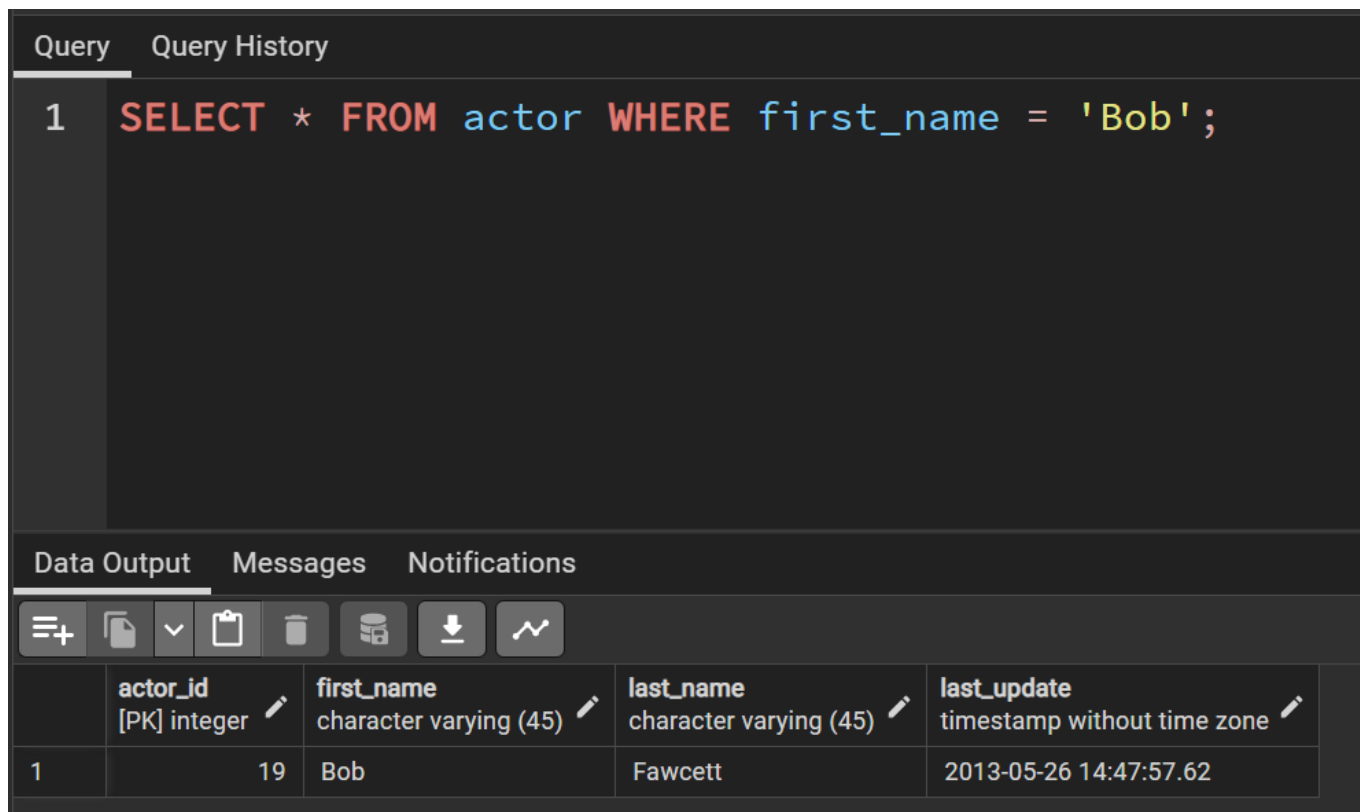
All of the standard operators are used:

= for equal to

\>= for greater or equal to

!= not equal to

and so on



The screenshot shows a SQL IDE interface. At the top, there are tabs for 'Query' and 'Query History'. The 'Query' tab is active, displaying a SQL query: `1 SELECT * FROM actor WHERE first_name = 'Bob';`. Below the query editor, there are tabs for 'Data Output', 'Messages', and 'Notifications'. The 'Data Output' tab is active, showing a table of results. The table has four columns: 'actor_id' (integer, primary key), 'first_name' (character varying (45)), 'last_name' (character varying (45)), and 'last_update' (timestamp without time zone). The first row of data shows actor_id 19, first_name 'Bob', last_name 'Fawcett', and last_update '2013-05-26 14:47:57.62'.

	actor_id [PK] integer	first_name character varying (45)	last_name character varying (45)	last_update timestamp without time zone
1	19	Bob	Fawcett	2013-05-26 14:47:57.62

Only selecting the column which contains the item you're looking for would probably not be very useful since probably you want the corresponding info in *other* columns, i.e. last name, actor id, etc. **So, probably don't do this:**

Query

Query History

1

SELECT first_name FROM actor WHERE first_name = 'Bob';

Data Output

Messages

Notifications

≡+

📄

▼

📋

🗑️

🗄️

⬇️

📈

	<div>first_name</div> <div>character varying (45) 🔒</div>
1	Bob

Also note: The name of the item you're looking for (shown in yellow) goes in single quotes. The column name and item name are both case-sensitive.

Working with multiple conditions:

Query

Query History

1

SELECT * from film WHERE rental_rate = .99 AND rating = 'NC-17';

Data Output

Messages

Notifications

≡+

📄

▼

📋

🗑️

🗄️

⬇️

📈

		release_year integer	language_id smallint	rental_duration smallint	rental_rate numeric (4,2)	length smallint	replacement_cost numeric (5,2)	rating mpaa_rating	last_update timestamp v
1		2006	1	6	0.99	94	23.99	NC-17	2013-05-26
2	r in A Shark T...	2006	1	7	0.99	179	12.99	NC-17	2013-05-26
3		2006	1	6	0.99	62	29.99	NC-17	2013-05-26
4		2006	1	6	0.99	68	25.99	NC-17	2013-05-26
5		2006	1	6	0.99	53	27.99	NC-17	2013-05-26

I guess that the .99 doesn't have to be in \ ' ' since it's a number not a string.

Some more examples:

"Let's say a customer walks in asking for someone named Jared"

Query

Query History

1

SELECT * FROM customer

2

WHERE first_name = 'Jared';

Data Output

Messages

Notifications

	customer_id [PK] integer	store_id smallint	first_name character varying (45)	last_name character varying (45)	email character varying (50)	address_id smallint	activebool boolean	create_date date
1	524	1	Jared	Ely	jared.ely@sakilacustomer.org	530	true	2006-02-1

Breaking this into two lines is a good idea for the sake of clarity.

Finding examples of films which are both over \$4 to rent and have a replacment cost greater than or equal to \$19.99:

Query

Query History

1

SELECT * FROM film

2

WHERE rental_rate > 4

3

AND replacement_cost >= 19.99;

Data Output

Messages

Notifications

</

Not every column may be needed. Let's say we only want the title of these movies:

Query

Query History

1

SELECT title

FROM film

2

WHERE rental_rate > 4

3

AND replacement_cost >= 19.99;

Data Output

Messages

Notifications

≡+

	title character varying (255)	
1	Grosse Wonderful	
2	Airplane Sierra	
3	Aladdin Calendar	
4	Ali Forever	
5	Amelie Hellfighters	
6	Dying Maker	
Total rows: 173 of 173		Query complete 00:00:00.239

What if we want to know *how many* movies are \$4 to rent, \$19.99 to replace and have an 'R' rating?:

Query

Query History

1

SELECT Count(title) FROM film

2

WHERE rental_rate > 4

3

AND replacement_cost >= 19.99


4


AND rating = 'R';


Data Output


Messages


Notifications





















	count bigint	
1	34	

Although actually, this can be found without using COUNT:

Query

Query History

1

SELECT title FROM film

2

WHERE rental_rate > 4

3

AND replacement_cost >= 19.99

4

AND rating = 'R';

Data Output

Messages

Notifications

≡+

📄

▼

📋

🗑️

🗄️

⬇️

📈

	title character varying (255) 🔒
1	Grosse Wonderful
2	Amelie Hellfighters
3	Beast Hunchback
4	Brooklyn Desert
5	Bubble Grosse
6	Connecticut Tramp

Total rows: 34 of 34

Query complete 00:00:00.298

The count is shown by the number of rows. Again, specifying 'title' as the column is not needed, * works fine too.

OR

Query

Query History

1

2

SELECT title, rating FROM film

WHERE rating = 'R' or rating = 'PG-13';

Data Output

Messages

Notifications

≡+

📄

▼

📋

🗑️

🗄️

⬇️

📈

	<div>title</div> <div>character varying (255)</div> <div>🔒</div>	<div>rating</div> <div>mpaa_rating</div> <div>🔒</div>
1	Grosse Wonderful	R
2	Airport Pollock	R
3	Bright Encounters	PG-13
4	Airplane Sierra	PG-13
5	Alabama Devil	PG-13
6	Date Speed	R

Total rows: 418 of 418

Query complete 00:00:00.191

There are 418 movies which are either PG-13 or R-rated.

!=

Query

Query History

1

SELECT COUNT(*) FROM film


2


WHERE rating != 'R';


Data Output


Messages


Notifications





















	count bigint	
1	805	

805 movies have a rating other than 'R'.