Unit 7: Tools for data management

- Graphical user interface tools of the database management system
- Tables and data creation
- Record fetching
- Security options
- Performance
- Indices

Graphical tools

- There are graphical interfaces that allow the users to avoid the interactaction with the database system through text
- For instance, in Postgres pgAdmin is installed by default: pgadmin.org
- This tool is designed to meet the needs of both novice and expert users

Graphical tools

- PgAdmin provides a graphical interface that simplifies the creation, maintenance, and use of the database
- Compatible with Linux, Mac OS and Windows
- It fully supports the functionality of Postgres (queries, table creation and deletion, security management, etc.)
- It can be used by any type of user, although the tool is designed mainly for the use and convenience of administrators

Graphical tools

Features:

- Graphical display of the query plan
- Procedural language debugger
- Monitoring control panel
- Backup, restore and scan on demand
- Live SQL query tool with direct data edition

Graphical tools

Features (continued):

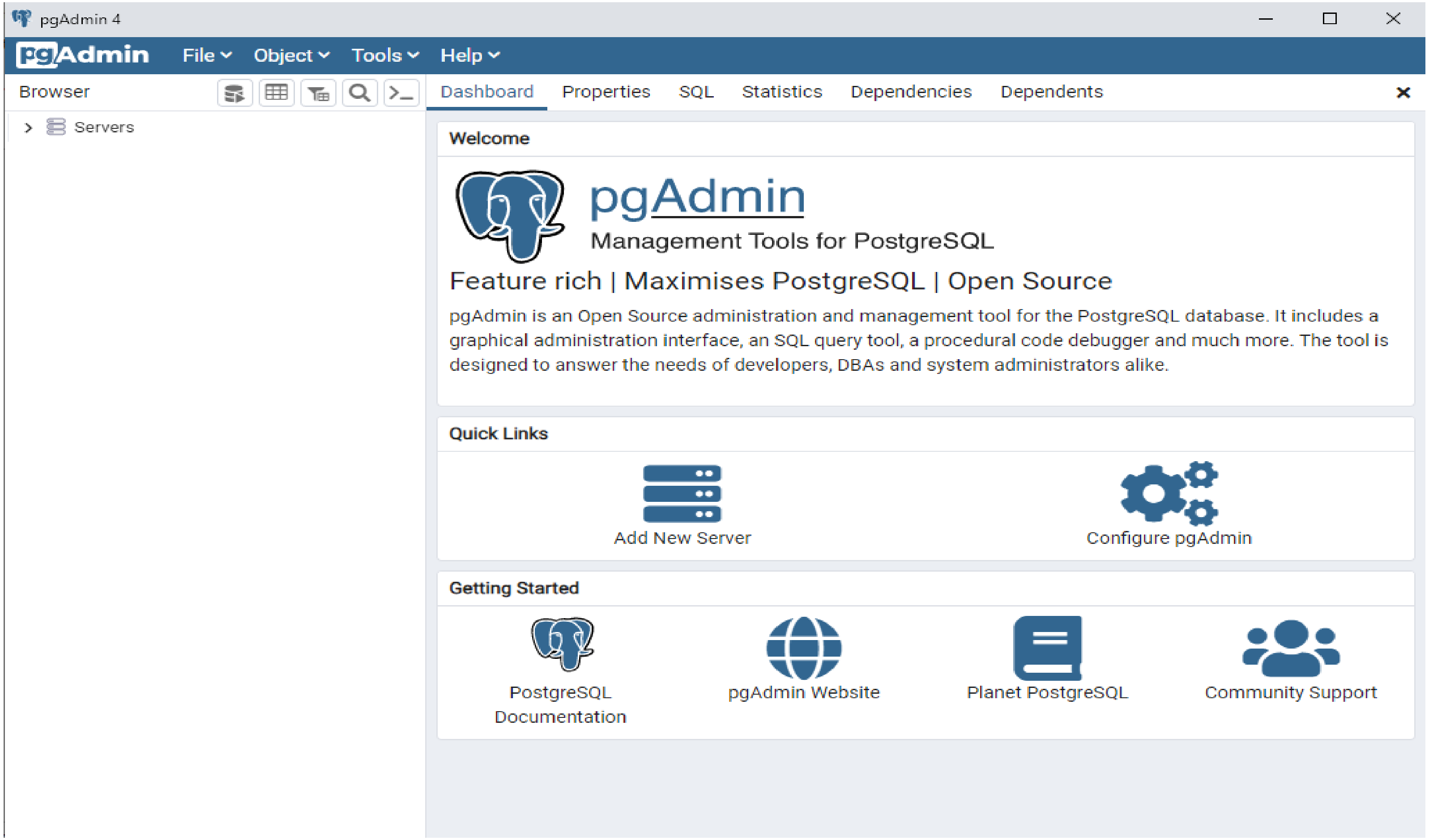
- SQL editor with syntax highlighting
- SQL batch job scheduling agent
- Context sensitive behavior
- Help and error messages
- Online help and information about its use

Graphical tools

- How to run:
  - pgAdmin4 tool
  - Web browser (connecting to the database server on localhost, specifying the Postgres port → 5432)
- Once connected to the server as a user (usually the administrator), operation with any database can be carried out

Graphical tools

Landing page / Initial screen



pgAdmin

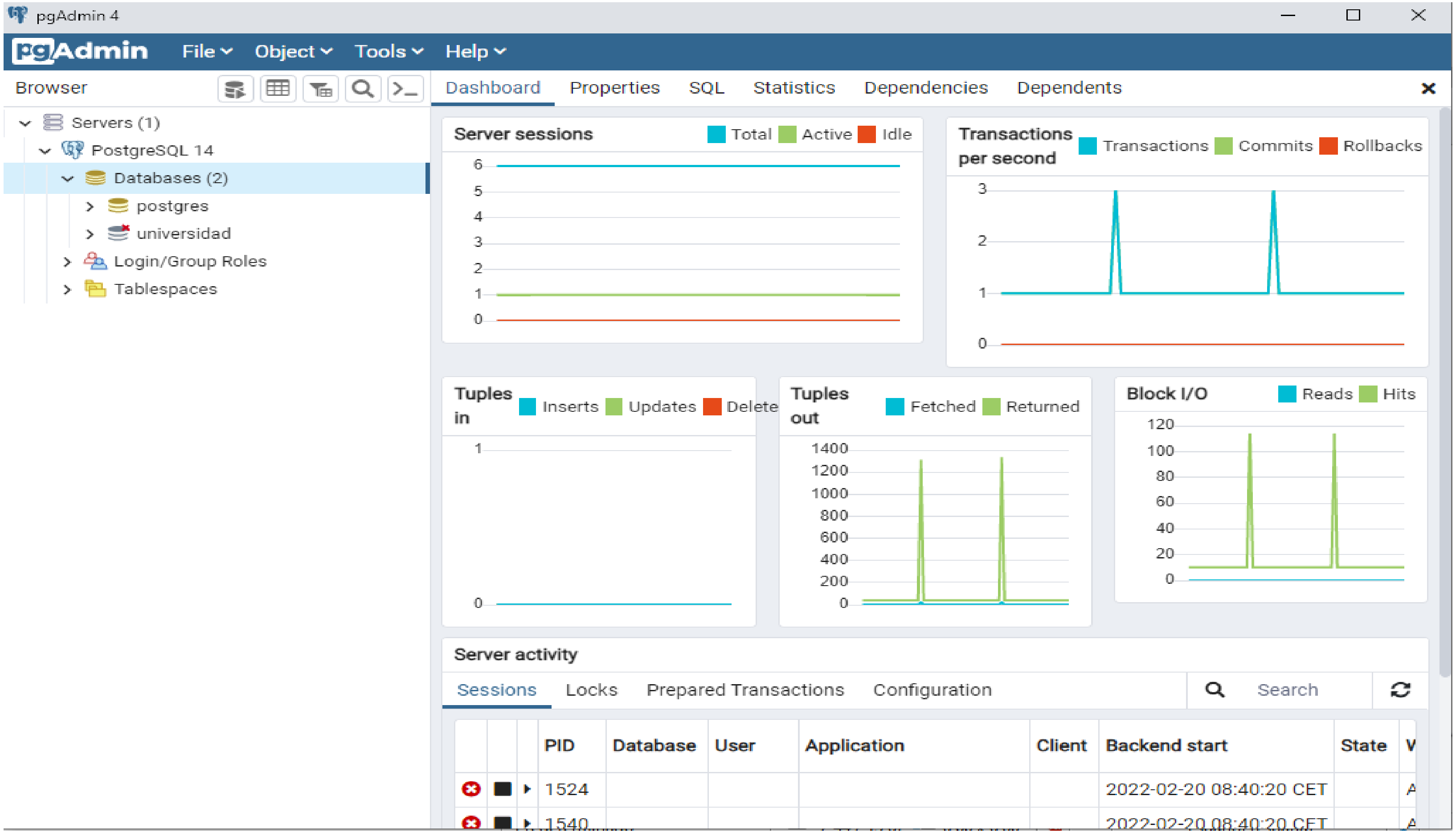
Management Tools for PostgreSQL

Feature rich | Maximises PostgreSQL | Open Source

pgAdmin is an Open Source administration and management tool for the PostgreSQL database. It includes a graphical administration interface, an SQL query tool, a procedural code debugger and much more. The tool is designed to answer the needs of developers, DBAs and system administrators alike.

Graphical tools

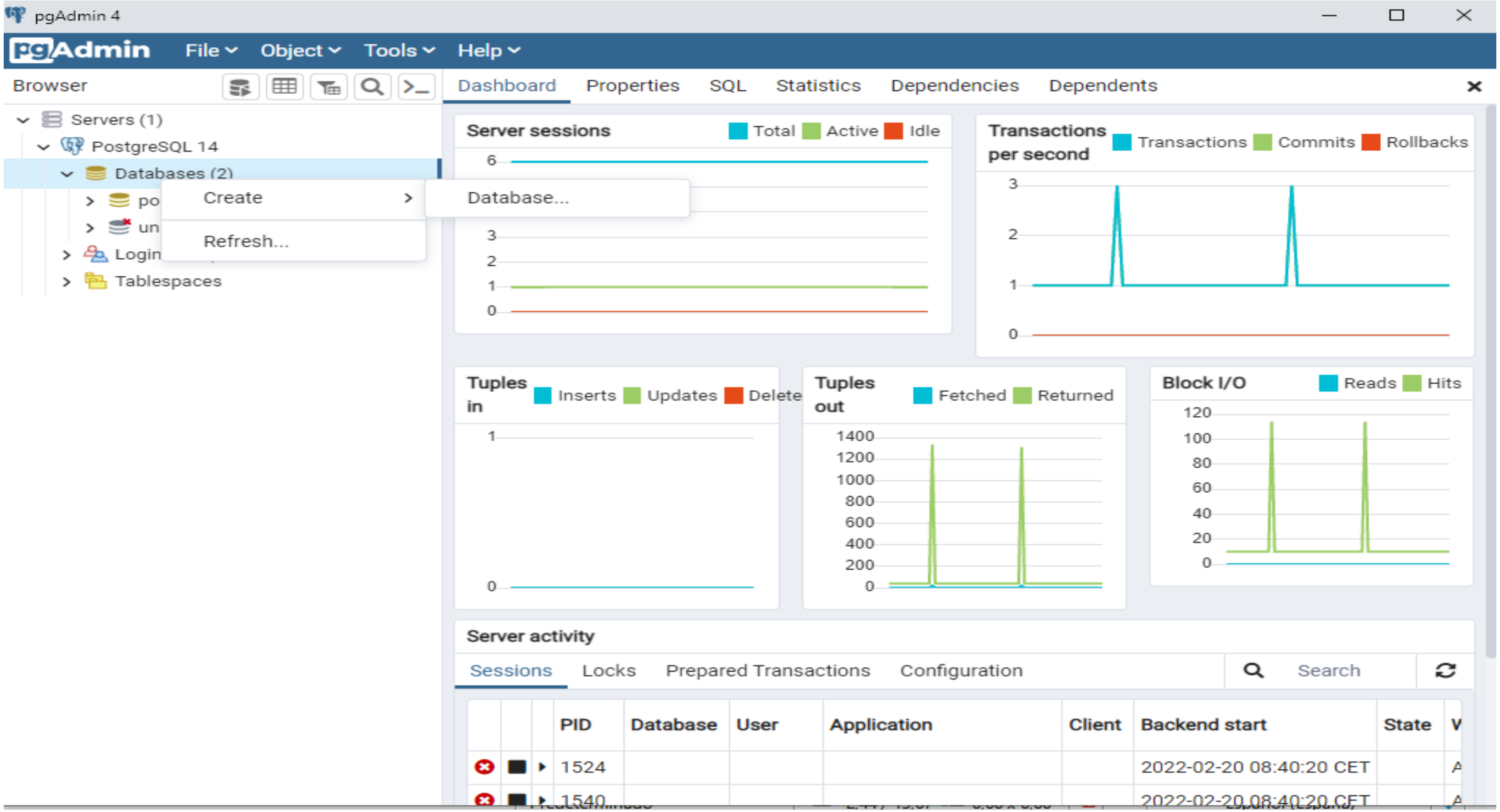
Overall looks



			PID	Database	User	Application	Client	Backend start	State	W
			1524					2022-02-20 08:40:20 CET		A
			1540					2022-02-20 08:40:20 CET		A

Creation of tables and data

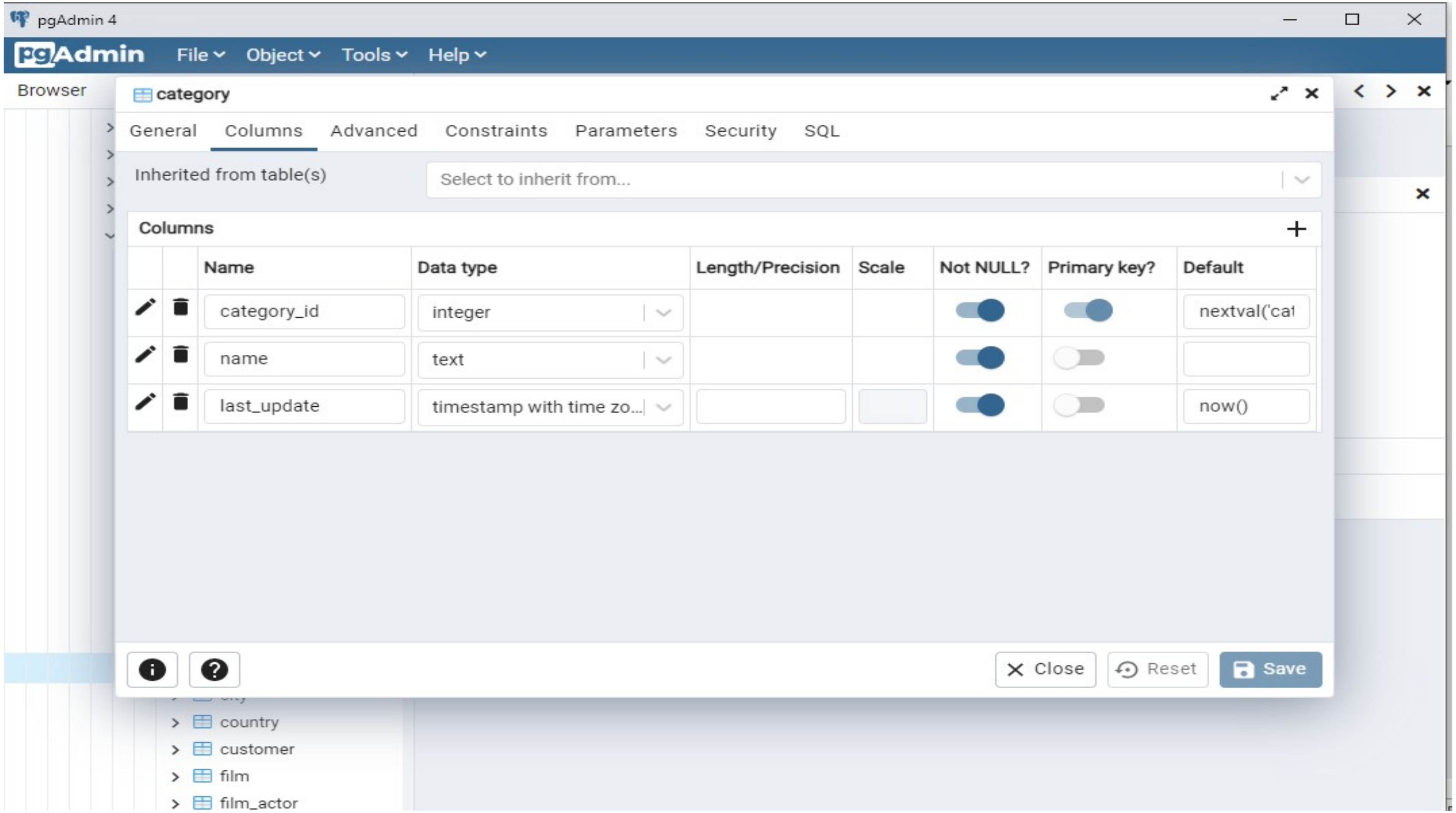
Creation of a database



			PID	Database	User	Application	Client	Backend start	State	W
			1524					2022-02-20 08:40:20 CET		A
			1540					2022-02-20 08:40:20 CET		A

Creation of tables and data

Columns and data types



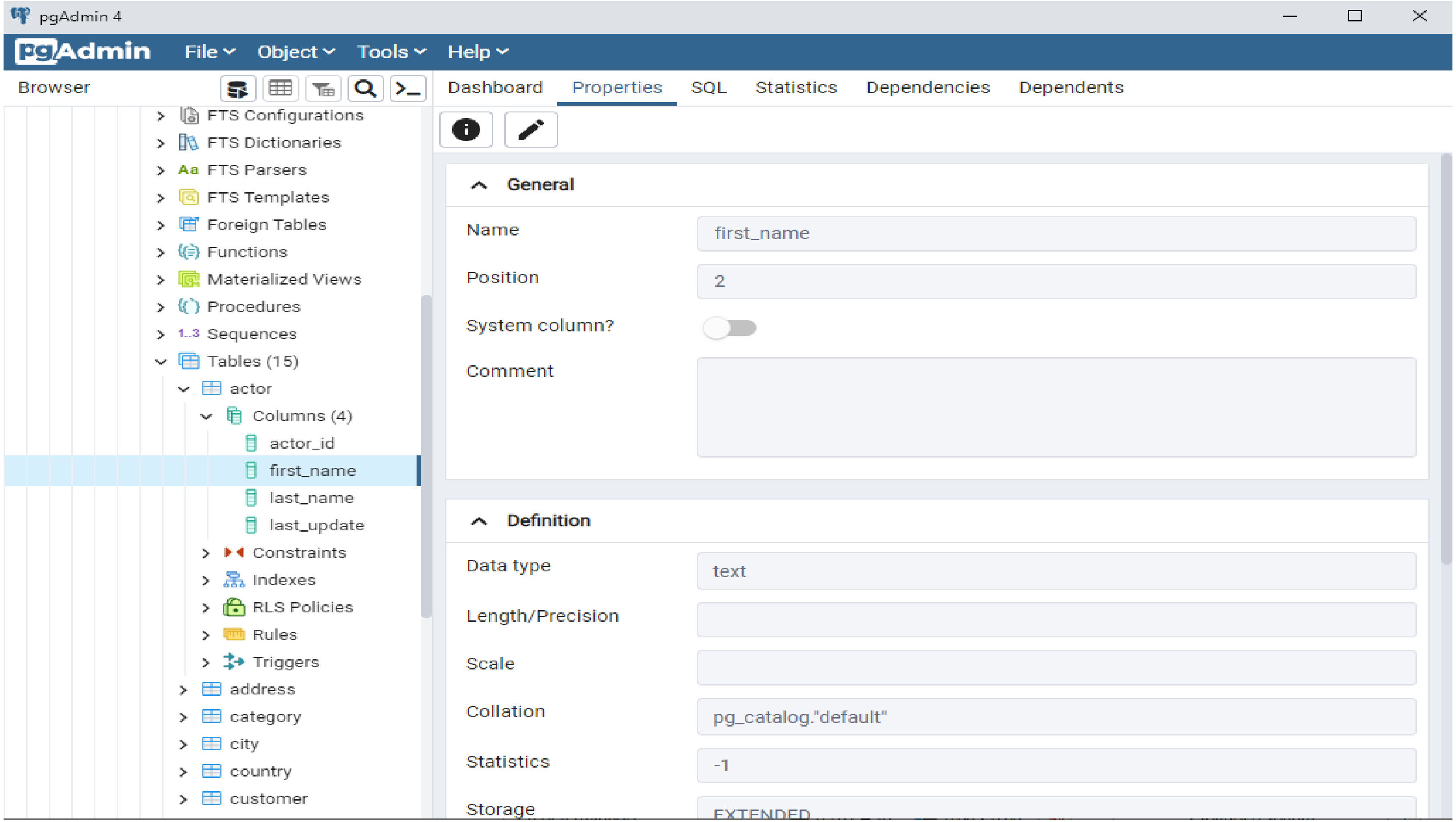
Inherited from table(s): Select to inherit from...

Columns

Name	Data type	Length/Precision	Scale	Not NULL?	Primary key?	Default
category_id	integer			●	●	nextval('cat
name	text			●	○	
last_update	timestamp with time zo...			●	○	now()

Creation of tables and data

Checking and modifying existing fields



General

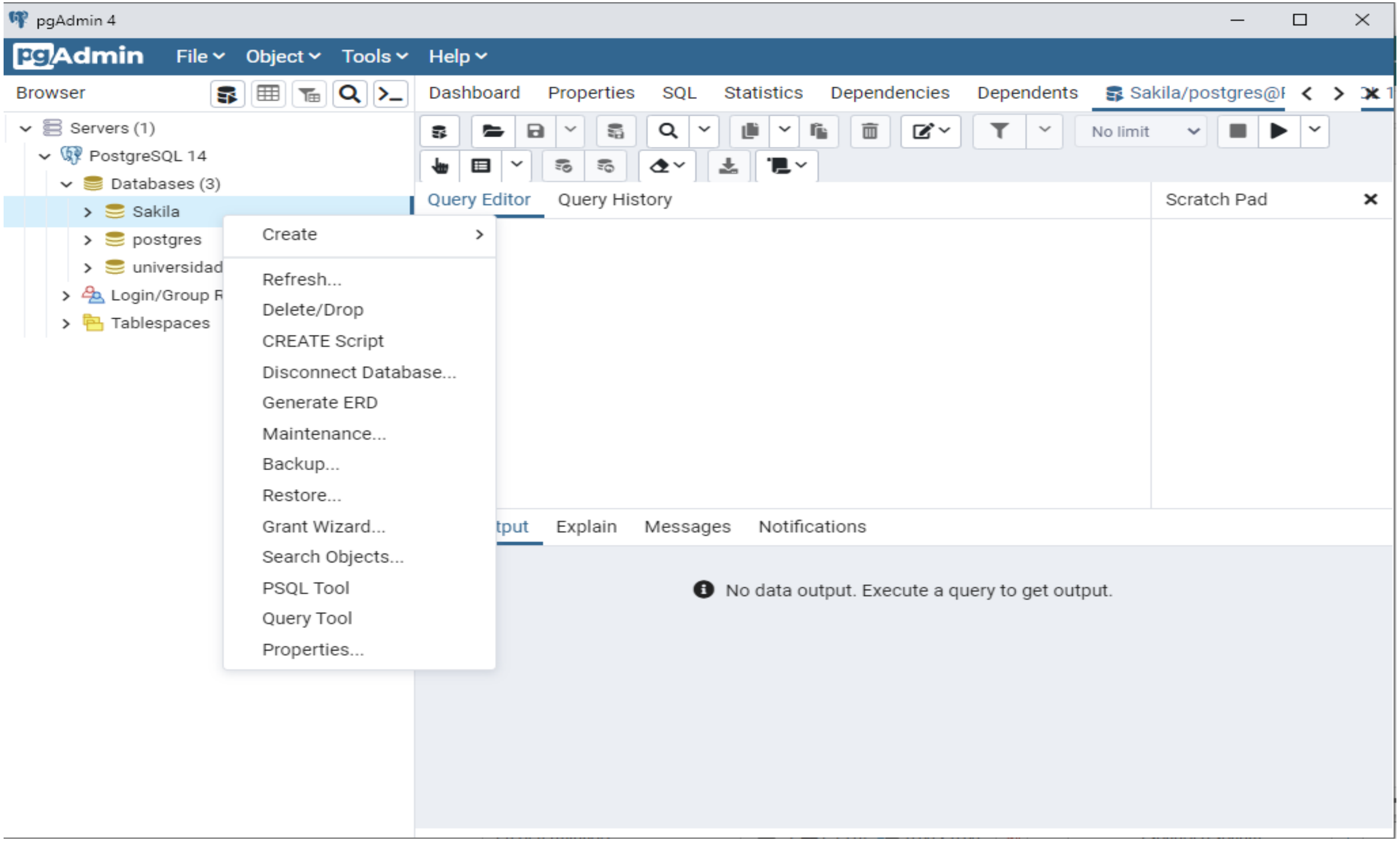
Name	first_name
Position	2
System column?	
Comment	

Definition

Data type	text
Length/Precision	
Scale	
Collation	pg_catalog."default"
Statistics	-1
Storage	EXTENDED

Creation of tables and data

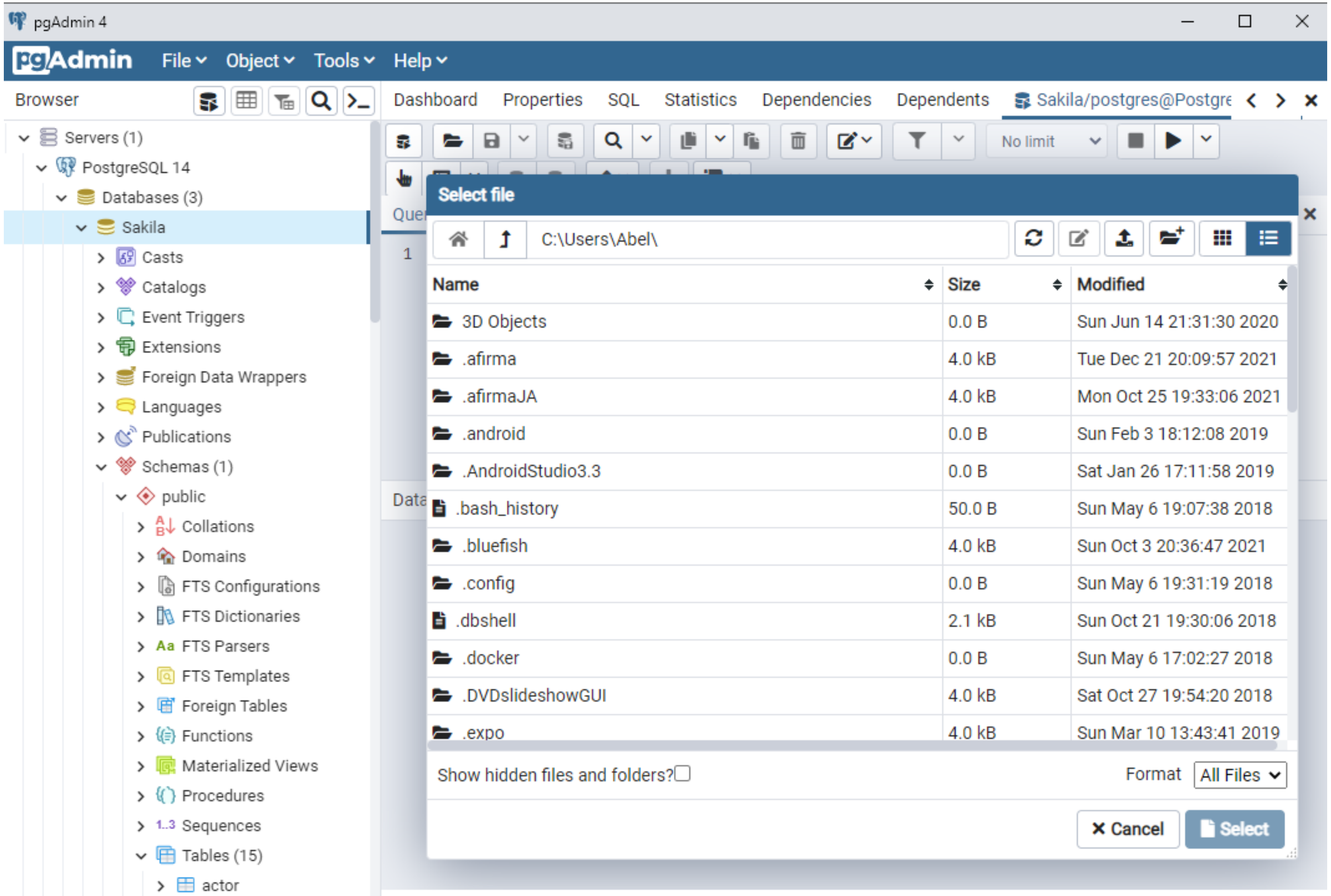
Create script
+
Load SQL file
+
Import / export (in tables)



No data output. Execute a query to get output.

Creation of tables and data

Load SQL script file



Select file

C:\Users\Abel\

Name	Size	Modified
3D Objects	0.0 B	Sun Jun 14 21:31:30 2020
.afirma	4.0 kB	Tue Dec 21 20:09:57 2021
.afirmaJA	4.0 kB	Mon Oct 25 19:33:06 2021
.android	0.0 B	Sun Feb 3 18:12:08 2019
.AndroidStudio3.3	0.0 B	Sat Jan 26 17:11:58 2019
.bash_history	50.0 B	Sun May 6 19:07:38 2018
.bluefish	4.0 kB	Sun Oct 3 20:36:47 2021
.config	0.0 B	Sun May 6 19:31:19 2018
.dbshell	2.1 kB	Sun Oct 21 19:30:06 2018
.docker	0.0 B	Sun May 6 17:02:27 2018
.DVDslideshowGUI	4.0 kB	Sat Oct 27 19:54:20 2018
.expo	4.0 kB	Sun Mar 10 13:43:41 2019

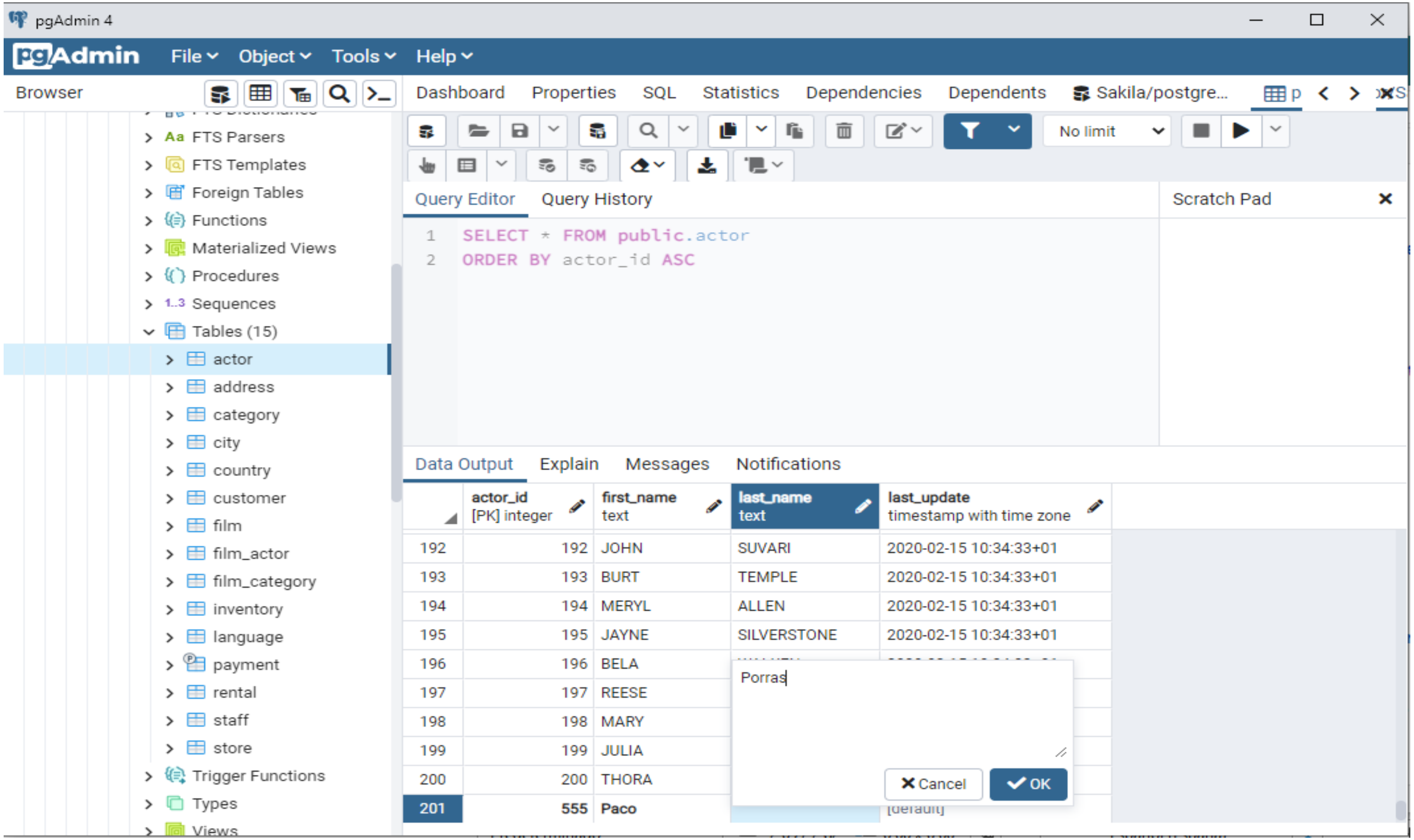
Show hidden files and folders?□

Format All Files

✕ Cancel | Select

Creation of tables and data

Add / delete data

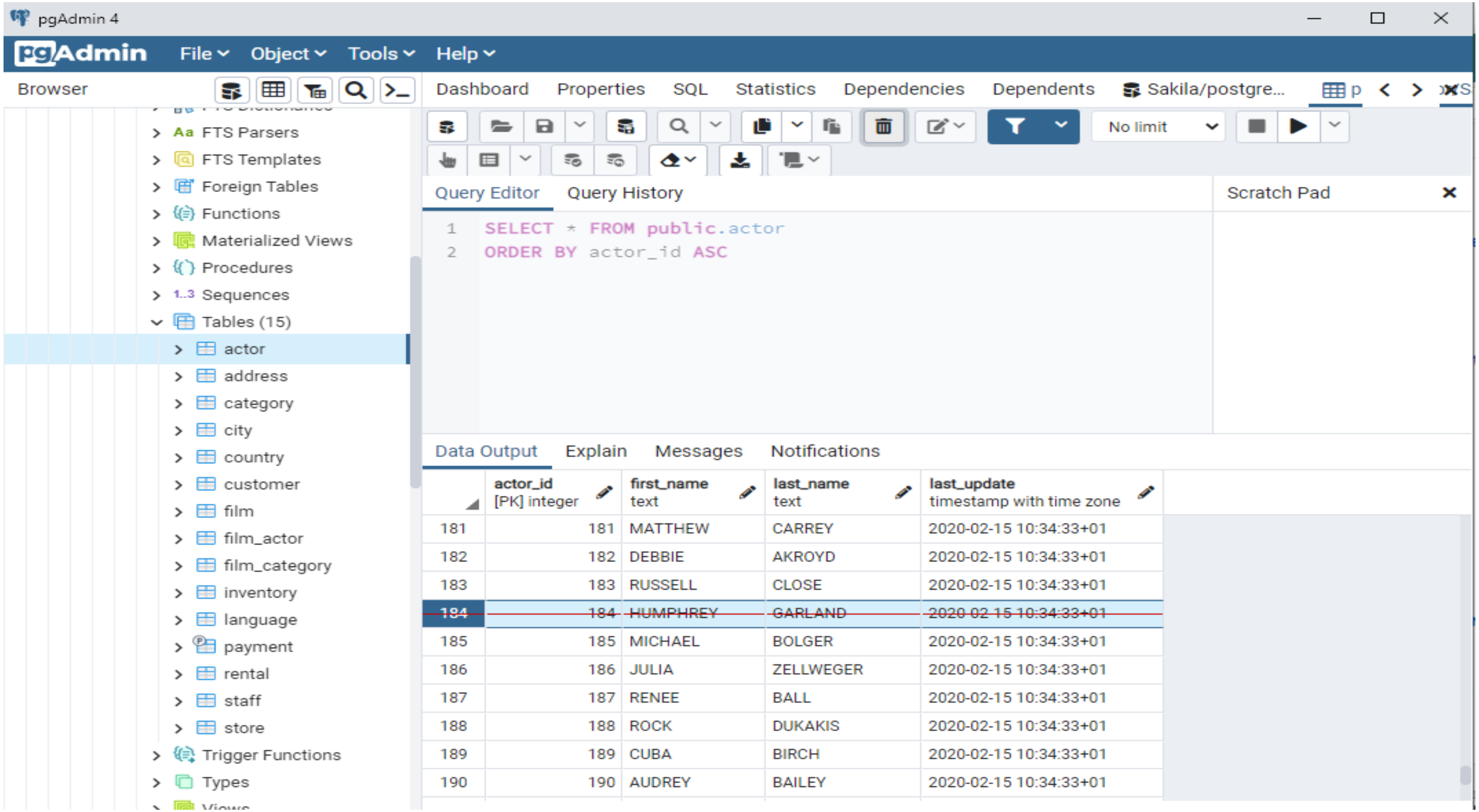


```
SELECT * FROM public.actor
ORDER BY actor_id ASC
```

	actor_id [PK] integer	first_name text	last_name text	last_update timestamp with time zone
192	192	JOHN	SUVARI	2020-02-15 10:34:33+01
193	193	BURT	TEMPLE	2020-02-15 10:34:33+01
194	194	MERYL	ALLEN	2020-02-15 10:34:33+01
195	195	JAYNE	SILVERSTONE	2020-02-15 10:34:33+01
196	196	BELA		
197	197	REESE		
198	198	MARY		
199	199	JULIA		
200	200	THORA		
201	555	Paco		[default]

Creation of tables and data

Add / delete data

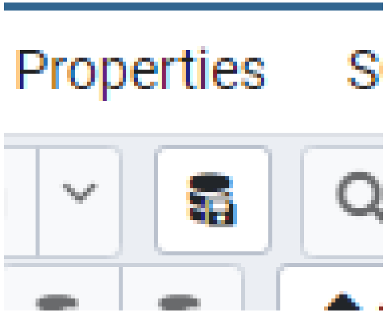


```sql
SELECT * FROM public.actor
ORDER BY actor_id ASC
```

	actor_id [PK] integer	first_name text	last_name text	last_update timestamp with time zone
181	181	MATTHEW	CARREY	2020-02-15 10:34:33+01
182	182	DEBBIE	AKROYD	2020-02-15 10:34:33+01
183	183	RUSSELL	CLOSE	2020-02-15 10:34:33+01
184	~~184~~	~~HUMPHREY~~	~~GARLAND~~	~~2020-02-15 10:34:33+01~~
185	185	MICHAEL	BOLGER	2020-02-15 10:34:33+01
186	186	JULIA	ZELLWEGER	2020-02-15 10:34:33+01
187	187	RENEE	BALL	2020-02-15 10:34:33+01
188	188	ROCK	DUKAKIS	2020-02-15 10:34:33+01
189	189	CUBA	BIRCH	2020-02-15 10:34:33+01
190	190	AUDREY	BAILEY	2020-02-15 10:34:33+01

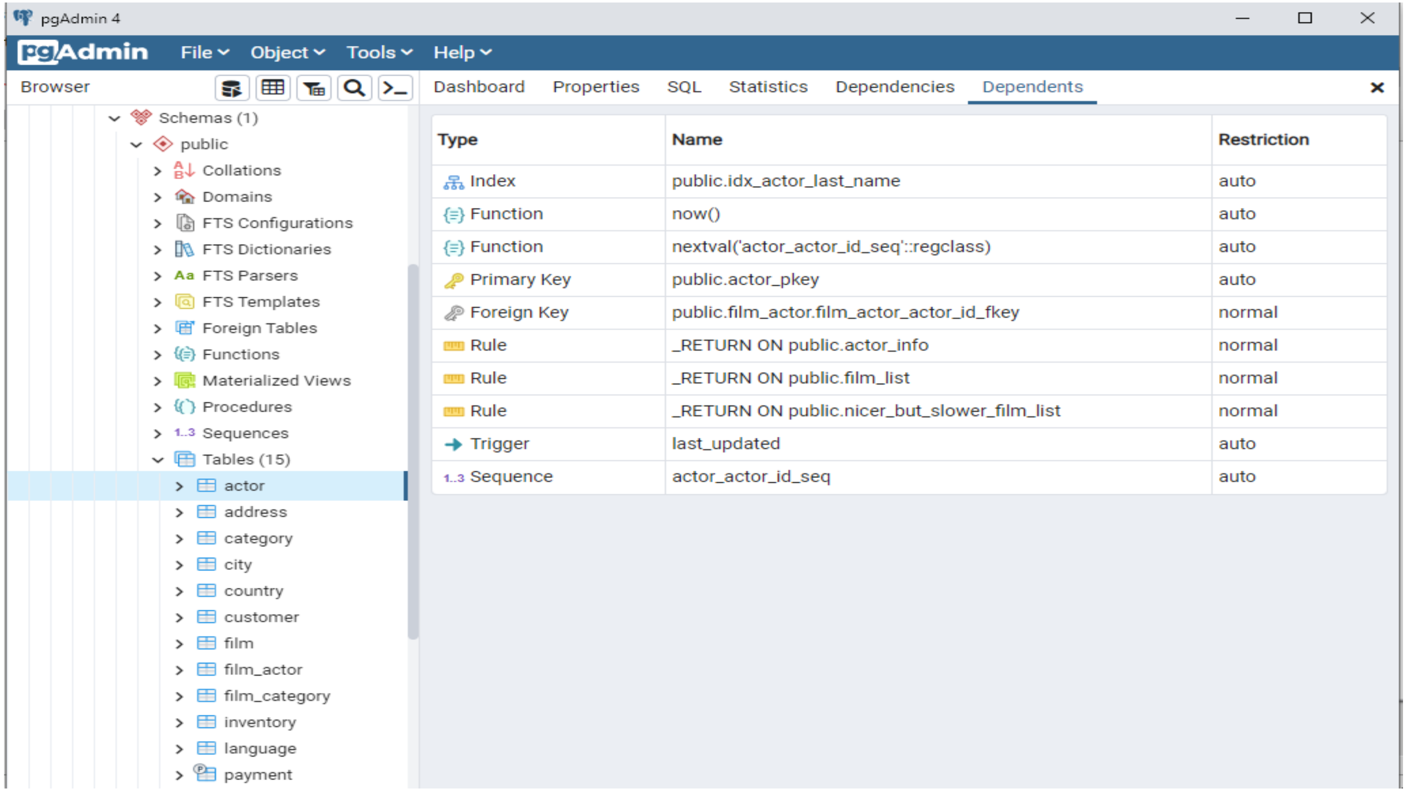
Creation of tables and data

- In order to actually get the new data, updates or deletion of rows, it is necessary to confirm the operation
- This is done through the F6 button or by clicking on the corresponding icon



Record selection

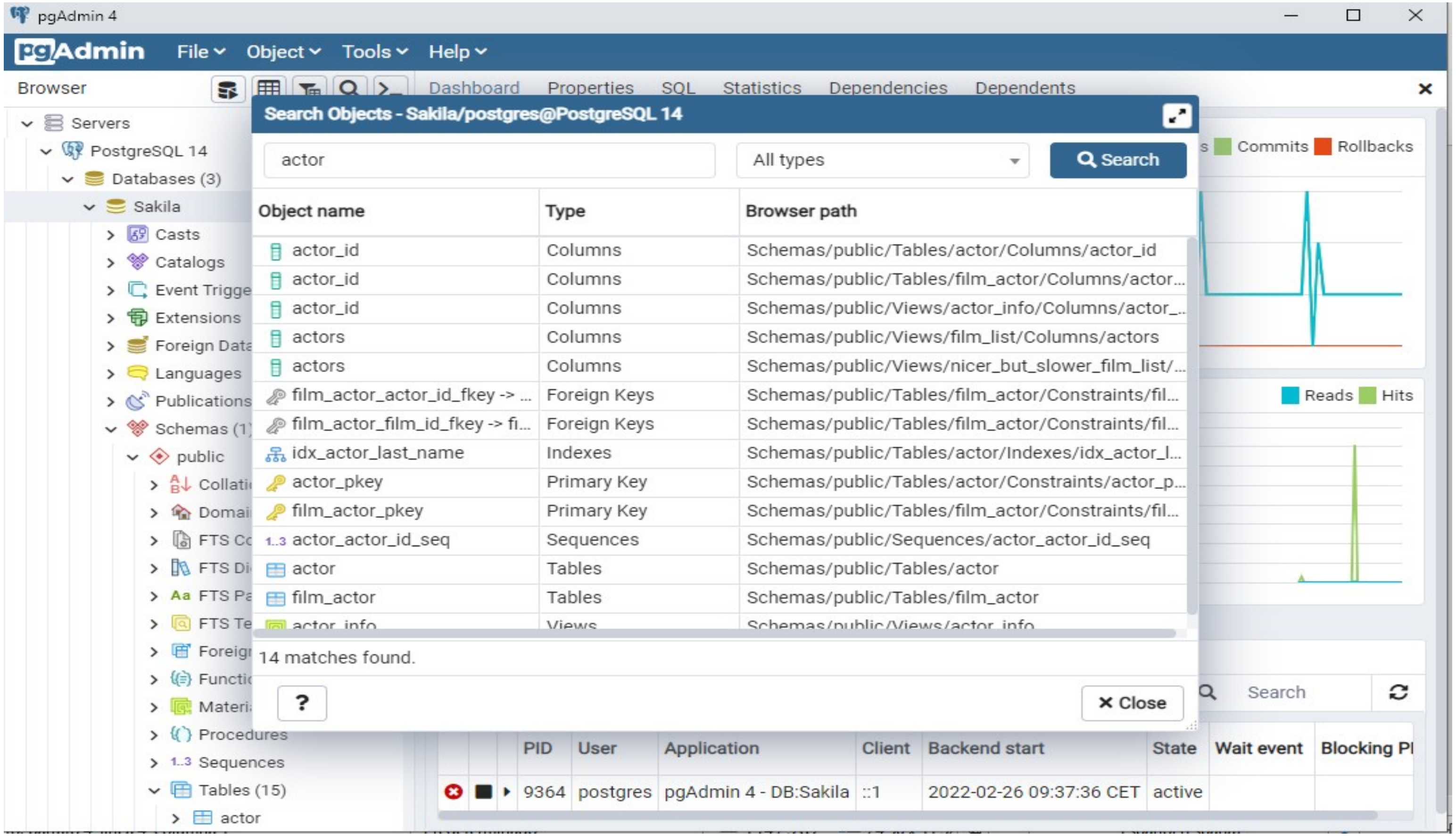
- Choice of data source:
  - Boards
  - Views



Type	Name	Restriction
Index	public.idx_actor_last_name	auto
Function	now()	auto
Function	nextval('actor_actor_id_seq'::regclass)	auto
Primary Key	public.actor_pkey	auto
Foreign Key	public.film_actor.film_actor_actor_id_fkey	normal
Rule	_RETURN ON public.actor_info	normal
Rule	_RETURN ON public.film_list	normal
Rule	_RETURN ON public.nicer_but_slower_film_list	normal
Trigger	last_updated	auto
Sequence	actor_actor_id_seq	auto

Record selection

- Looking for objects:



actor

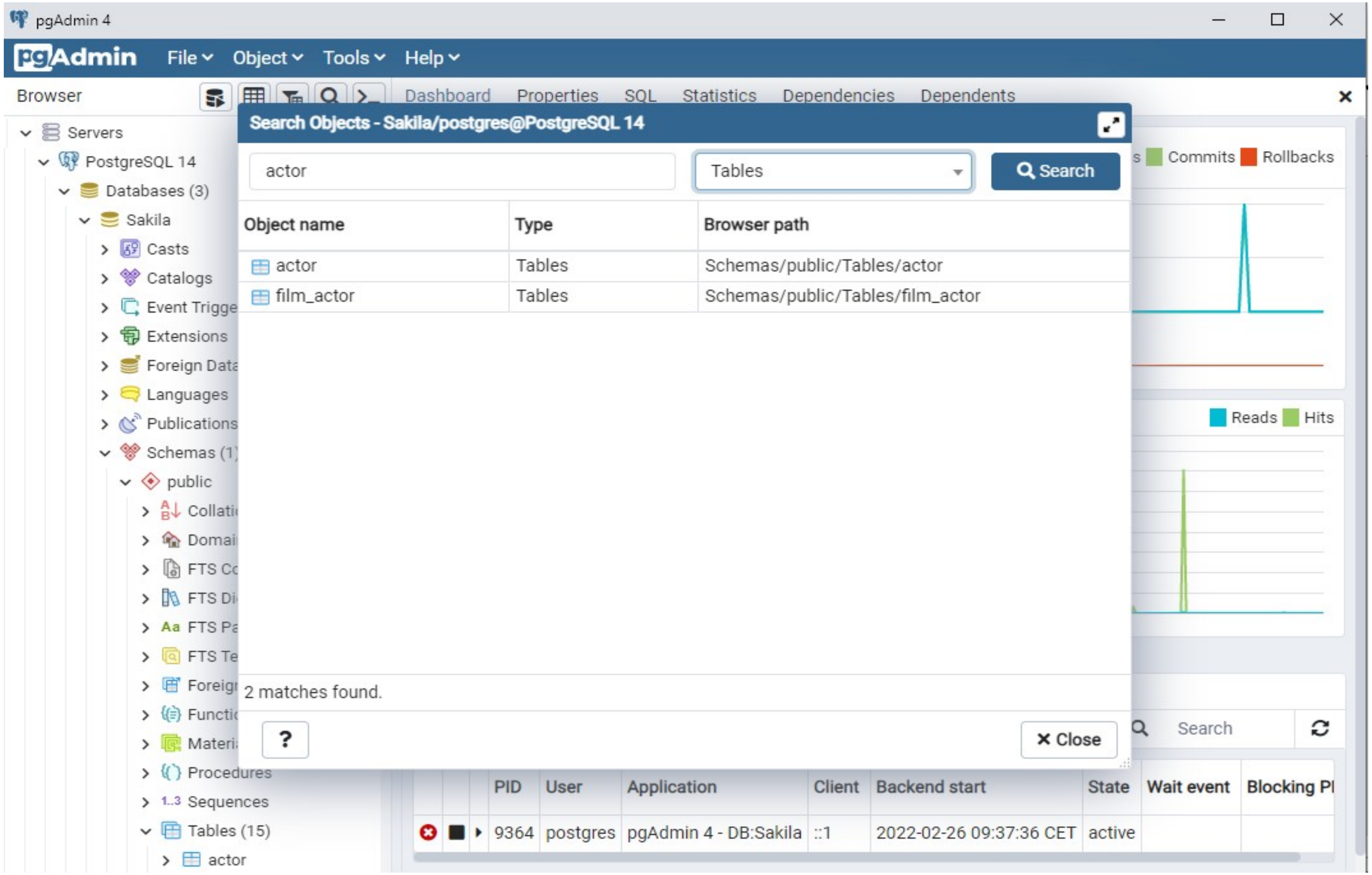
Object name	Type	Browser path
actor_id	Columns	Schemas/public/Tables/actor/Columns/actor_id
actor_id	Columns	Schemas/public/Tables/film_actor/Columns/actor...
actor_id	Columns	Schemas/public/Views/actor_info/Columns/actor_...
actors	Columns	Schemas/public/Views/film_list/Columns/actors
actors	Columns	Schemas/public/Views/nicer_but_slower_film_list/...
film_actor_actor_id_fkey -> ...	Foreign Keys	Schemas/public/Tables/film_actor/Constraints/fil...
film_actor_film_id_fkey -> fi...	Foreign Keys	Schemas/public/Tables/film_actor/Constraints/fil...
idx_actor_last_name	Indexes	Schemas/public/Tables/actor/Indexes/idx_actor_l...
actor_pkey	Primary Key	Schemas/public/Tables/actor/Constraints/actor_p...
film_actor_pkey	Primary Key	Schemas/public/Tables/film_actor/Constraints/fil...
actor_actor_id_seq	Sequences	Schemas/public/Sequences/actor_actor_id_seq
actor	Tables	Schemas/public/Tables/actor
film_actor	Tables	Schemas/public/Tables/film_actor
actor_info	Views	Schemas/public/Views/actor_info

14 matches found.

PID	User	Application	Client	Backend start	State	Wait event	Blocking PI
9364	postgres	pgAdmin 4 - DB:Sakila	::1	2022-02-26 09:37:36 CET	active		

Record selection

- Filter search:



Search Objects - Sakila/postgres@PostgreSQL 14

actor

Tables

Search

Object name	Type	Browser path
actor	Tables	Schemas/public/Tables/actor
film_actor	Tables	Schemas/public/Tables/film_actor

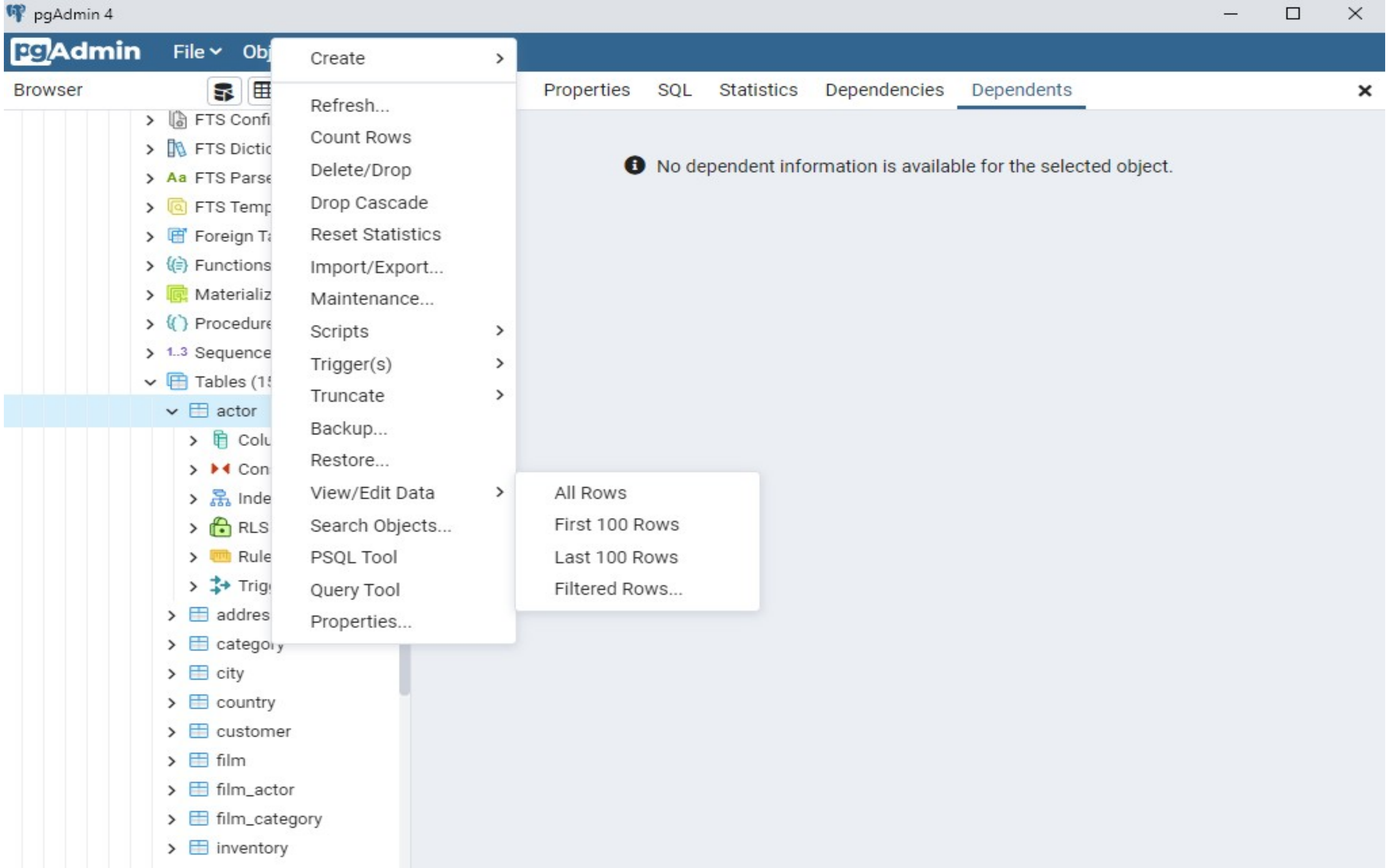
2 matches found.

? ✕ Close

			PID	User	Application	Client	Backend start	State	Wait event	Blocking PI
✖	■	▸	9364	postgres	pgAdmin 4 - DB:Sakila	::1	2022-02-26 09:37:36 CET	active		

Record selection

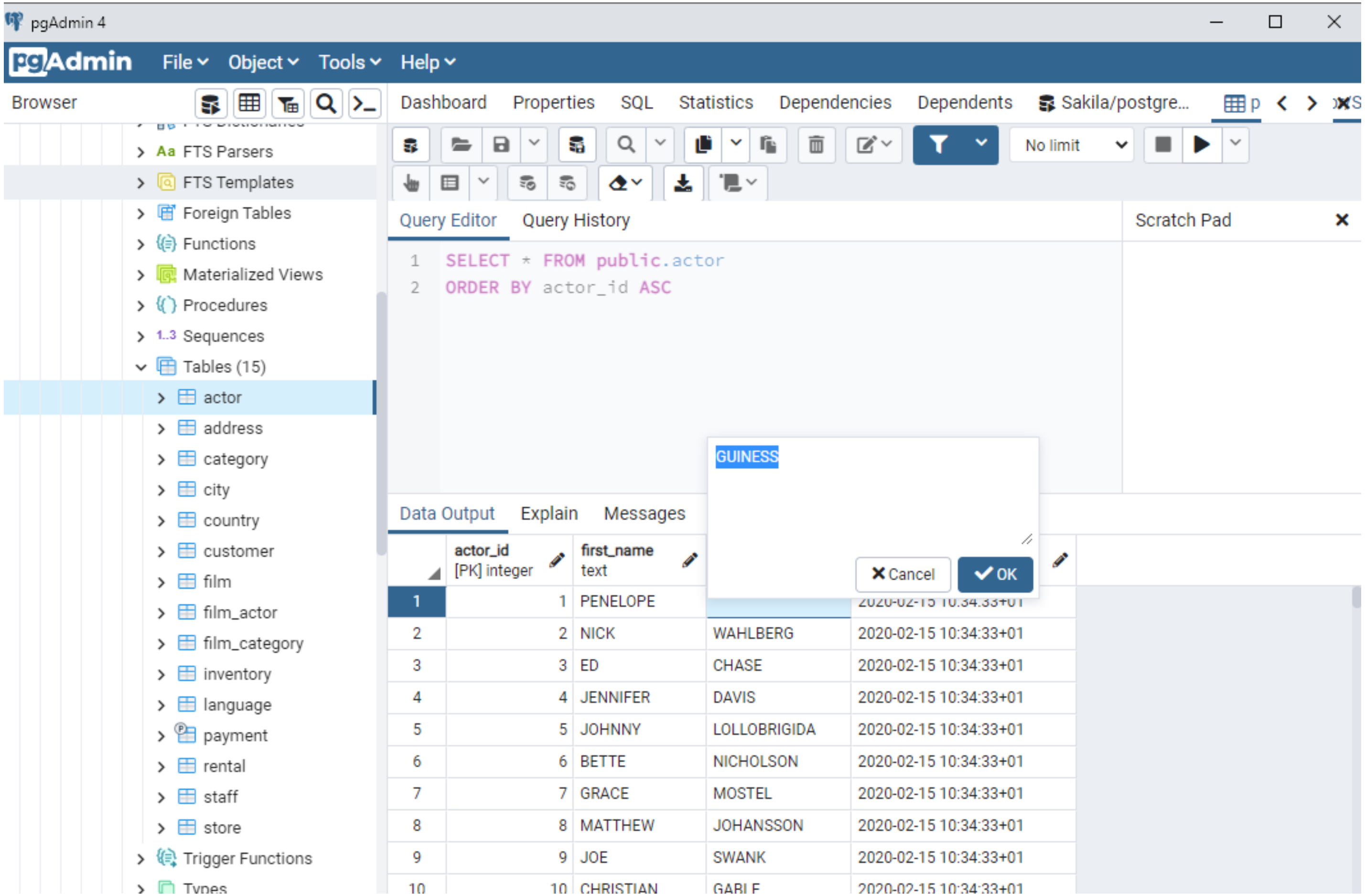
- First or last records:



No dependent information is available for the selected object.

Record selection

- Update data (similar to adding):



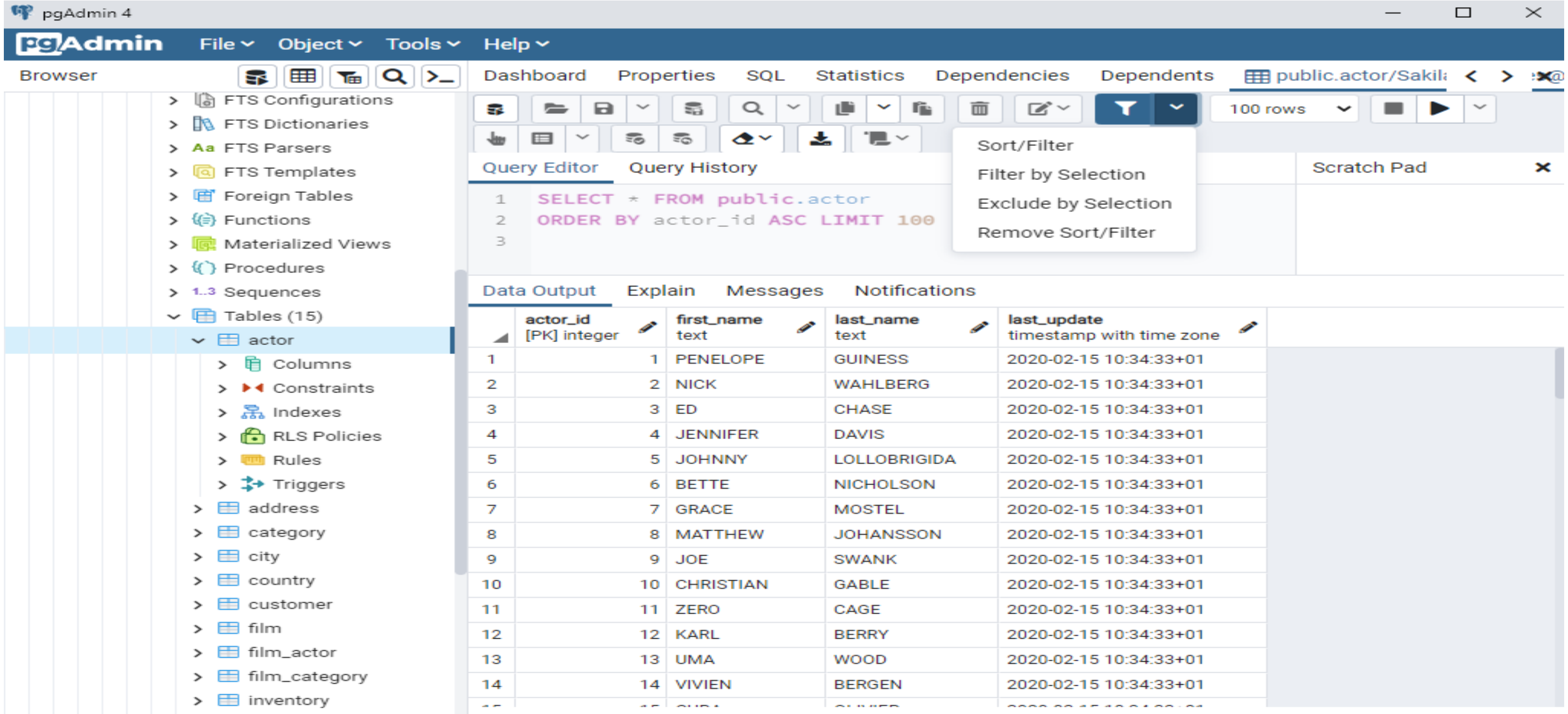
```
SELECT * FROM public.actor
ORDER BY actor_id ASC
```

GUINESS

	actor_id [PK] integer	first_name text		
1	1	PENELOPE		2020-02-15 10:34:33+01
2	2	NICK	WAHLBERG	2020-02-15 10:34:33+01
3	3	ED	CHASE	2020-02-15 10:34:33+01
4	4	JENNIFER	DAVIS	2020-02-15 10:34:33+01
5	5	JOHNNY	LOLLOBRIGIDA	2020-02-15 10:34:33+01
6	6	BETTE	NICHOLSON	2020-02-15 10:34:33+01
7	7	GRACE	MOSTEL	2020-02-15 10:34:33+01
8	8	MATTHEW	JOHANSSON	2020-02-15 10:34:33+01
9	9	JOE	SWANK	2020-02-15 10:34:33+01
10	10	CHRISTIAN	GABLE	2020-02-15 10:34:33+01

Record selection

- Filtering and sorting of records:

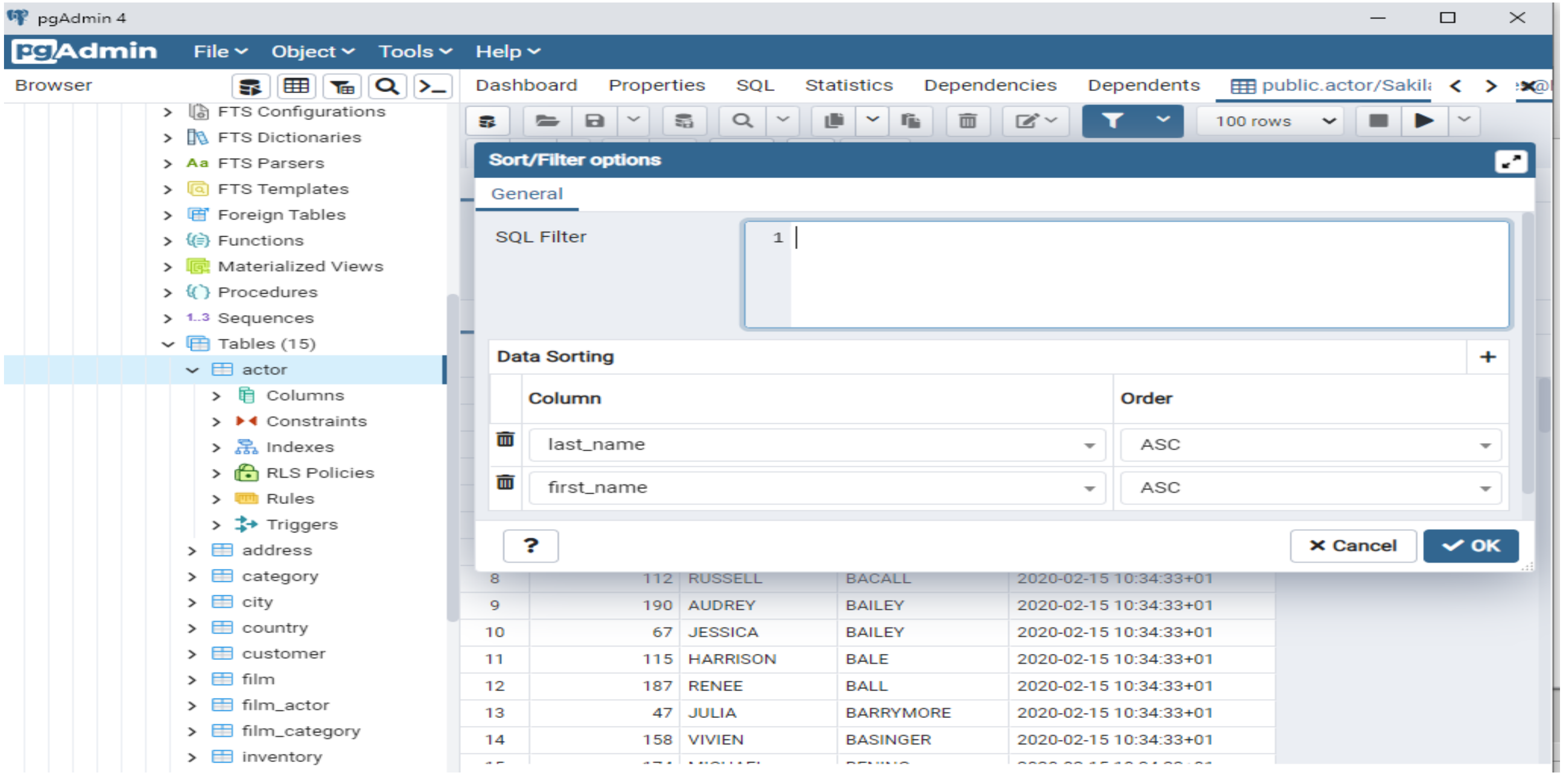


```
SELECT * FROM public.actor
ORDER BY actor_id ASC LIMIT 100
```

	actor_id [PK] integer	first_name text	last_name text	last_update timestamp with time zone
1	1	PENELOPE	GUINESS	2020-02-15 10:34:33+01
2	2	NICK	WAHLBERG	2020-02-15 10:34:33+01
3	3	ED	CHASE	2020-02-15 10:34:33+01
4	4	JENNIFER	DAVIS	2020-02-15 10:34:33+01
5	5	JOHNNY	LOLLOBRIGIDA	2020-02-15 10:34:33+01
6	6	BETTE	NICHOLSON	2020-02-15 10:34:33+01
7	7	GRACE	MOSTEL	2020-02-15 10:34:33+01
8	8	MATTHEW	JOHANSSON	2020-02-15 10:34:33+01
9	9	JOE	SWANK	2020-02-15 10:34:33+01
10	10	CHRISTIAN	GABLE	2020-02-15 10:34:33+01
11	11	ZERO	CAGE	2020-02-15 10:34:33+01
12	12	KARL	BERRY	2020-02-15 10:34:33+01
13	13	UMA	WOOD	2020-02-15 10:34:33+01
14	14	VIVIEN	BERGEN	2020-02-15 10:34:33+01

Record selection

- Filtering and sorting of records:



SQL Filter

```
1 |
```

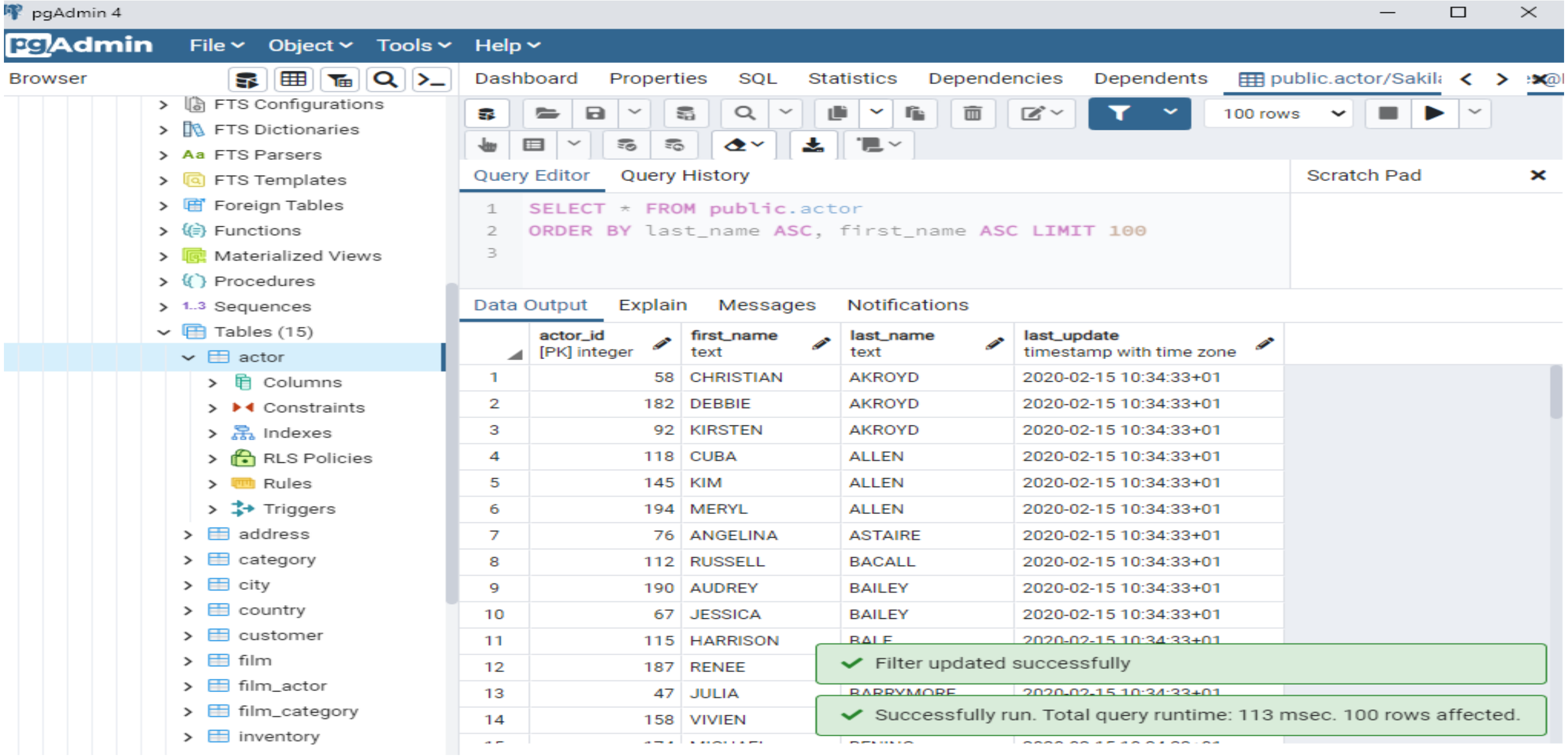
Data Sorting

Column	Order
last_name	ASC
first_name	ASC

8	112	RUSSELL	BACALL	2020-02-15 10:34:33+01
9	190	AUDREY	BAILEY	2020-02-15 10:34:33+01
10	67	JESSICA	BAILEY	2020-02-15 10:34:33+01
11	115	HARRISON	BALE	2020-02-15 10:34:33+01
12	187	RENEE	BALL	2020-02-15 10:34:33+01
13	47	JULIA	BARRYMORE	2020-02-15 10:34:33+01
14	158	VIVIEN	BASINGER	2020-02-15 10:34:33+01

Record selection

- Order of returned results:



```
SELECT * FROM public.actor
ORDER BY last_name ASC, first_name ASC LIMIT 100

```

	actor_id [PK] integer	first_name text	last_name text	last_update timestamp with time zone
1	58	CHRISTIAN	AKROYD	2020-02-15 10:34:33+01
2	182	DEBBIE	AKROYD	2020-02-15 10:34:33+01
3	92	KIRSTEN	AKROYD	2020-02-15 10:34:33+01
4	118	CUBA	ALLEN	2020-02-15 10:34:33+01
5	145	KIM	ALLEN	2020-02-15 10:34:33+01
6	194	MERYL	ALLEN	2020-02-15 10:34:33+01
7	76	ANGELINA	ASTAIRE	2020-02-15 10:34:33+01
8	112	RUSSELL	BACALL	2020-02-15 10:34:33+01
9	190	AUDREY	BAILEY	2020-02-15 10:34:33+01
10	67	JESSICA	BAILEY	2020-02-15 10:34:33+01
11	115	HARRISON	BALE	2020-02-15 10:34:33+01
12	187	RENEE		
13	47	JULIA	BARRYMORE	2020-02-15 10:34:33+01
14	158	VIVIEN		

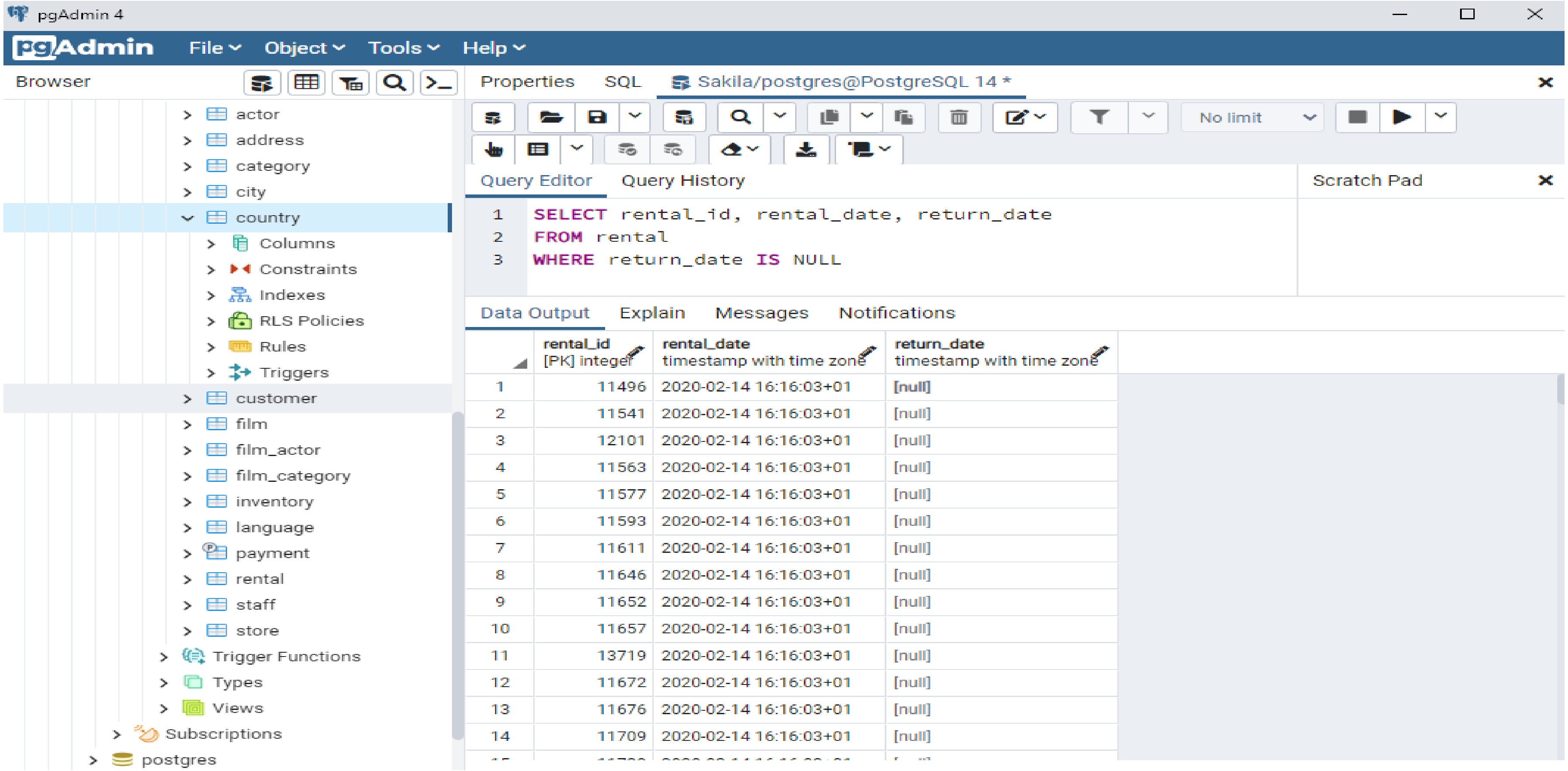
✓ Filter updated successfully

✓ Successfully run. Total query runtime: 113 msec. 100 rows affected.

Treatment of null values

- Null values require special treatment in relational databases, and specifically in SQL
- With the pgAdmin tool, identifying null values is easy
- However → how to enter them when importing data?
- https://www.pgadmin.org/docs/pgadmin4/development/import_export_data.html

Treatment of null values



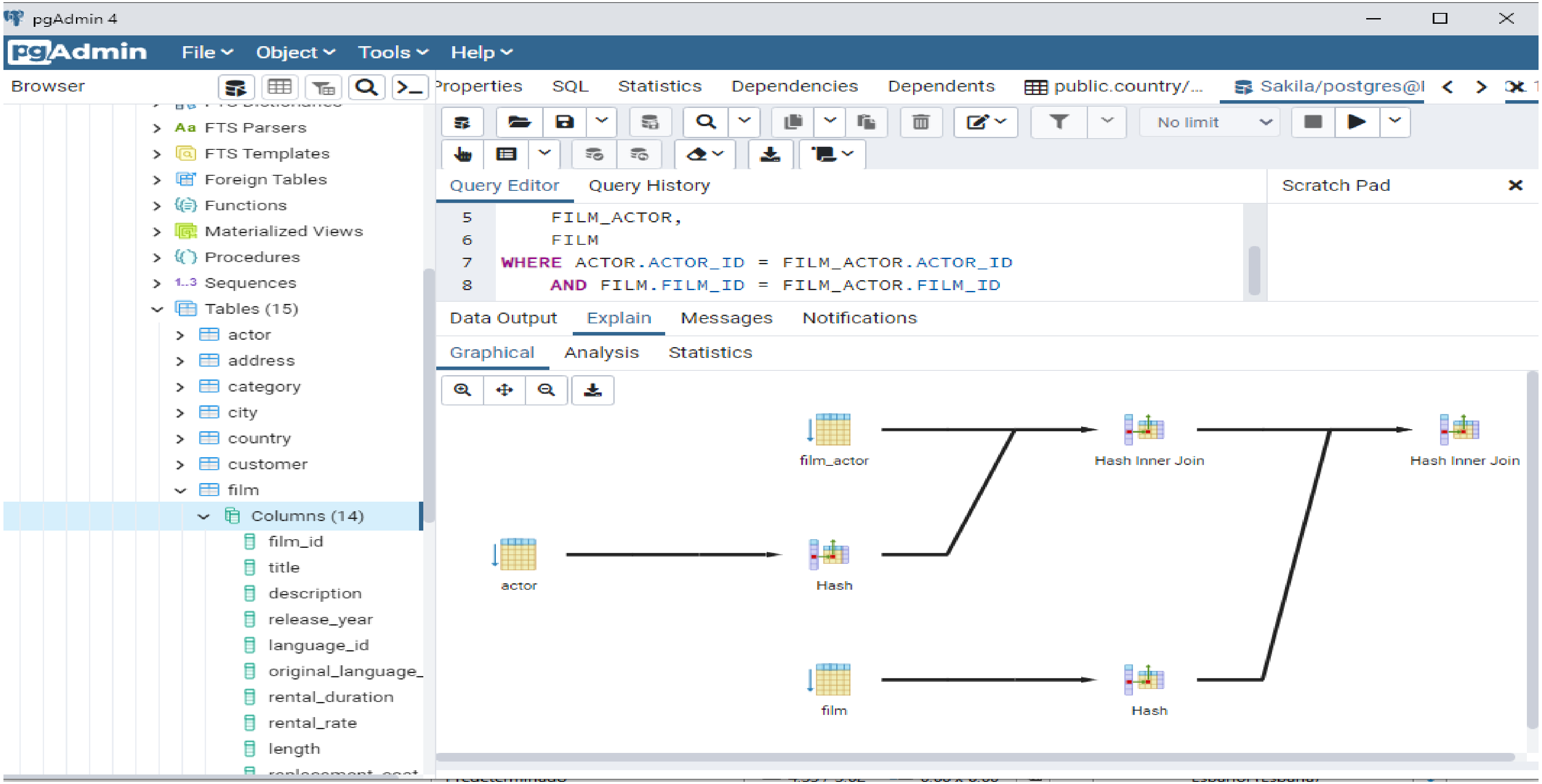
```sql
SELECT rental_id, rental_date, return_date
FROM rental
WHERE return_date IS NULL
```

	rental_id [PK] integer	rental_date timestamp with time zone	return_date timestamp with time zone
1	11496	2020-02-14 16:16:03+01	[null]
2	11541	2020-02-14 16:16:03+01	[null]
3	12101	2020-02-14 16:16:03+01	[null]
4	11563	2020-02-14 16:16:03+01	[null]
5	11577	2020-02-14 16:16:03+01	[null]
6	11593	2020-02-14 16:16:03+01	[null]
7	11611	2020-02-14 16:16:03+01	[null]
8	11646	2020-02-14 16:16:03+01	[null]
9	11652	2020-02-14 16:16:03+01	[null]
10	11657	2020-02-14 16:16:03+01	[null]
11	13719	2020-02-14 16:16:03+01	[null]
12	11672	2020-02-14 16:16:03+01	[null]
13	11676	2020-02-14 16:16:03+01	[null]
14	11709	2020-02-14 16:16:03+01	[null]

Performance

- pgAdmin provides a number of very handy visual performance options.
- Explain: option used to analyse the performance of a query
  https://www.postgresql.org/docs/current/sql-explain.html
- Some queries can be optimised by writing or executing them in an alternative way
- Other queries will require defining indexes on the tables to be more efficient

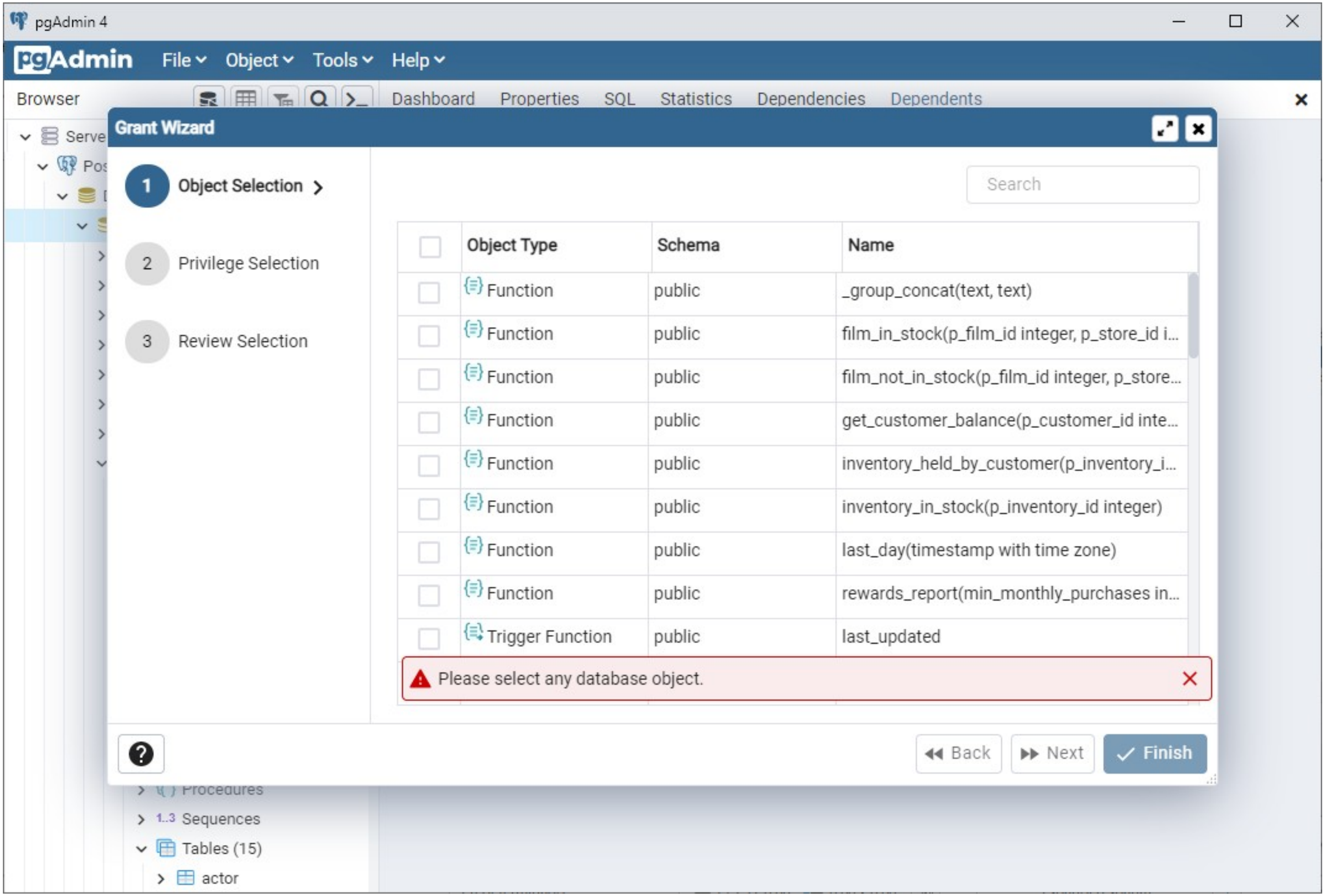
Performance



```
    FILM_ACTOR,
    FILM
WHERE ACTOR.ACTOR_ID = FILM_ACTOR.ACTOR_ID
    AND FILM.FILM_ID = FILM_ACTOR.FILM_ID
```

Security options

- Grant
  or withdraw
  permissions:



1 Object Selection

2 Privilege Selection

3 Review Selection

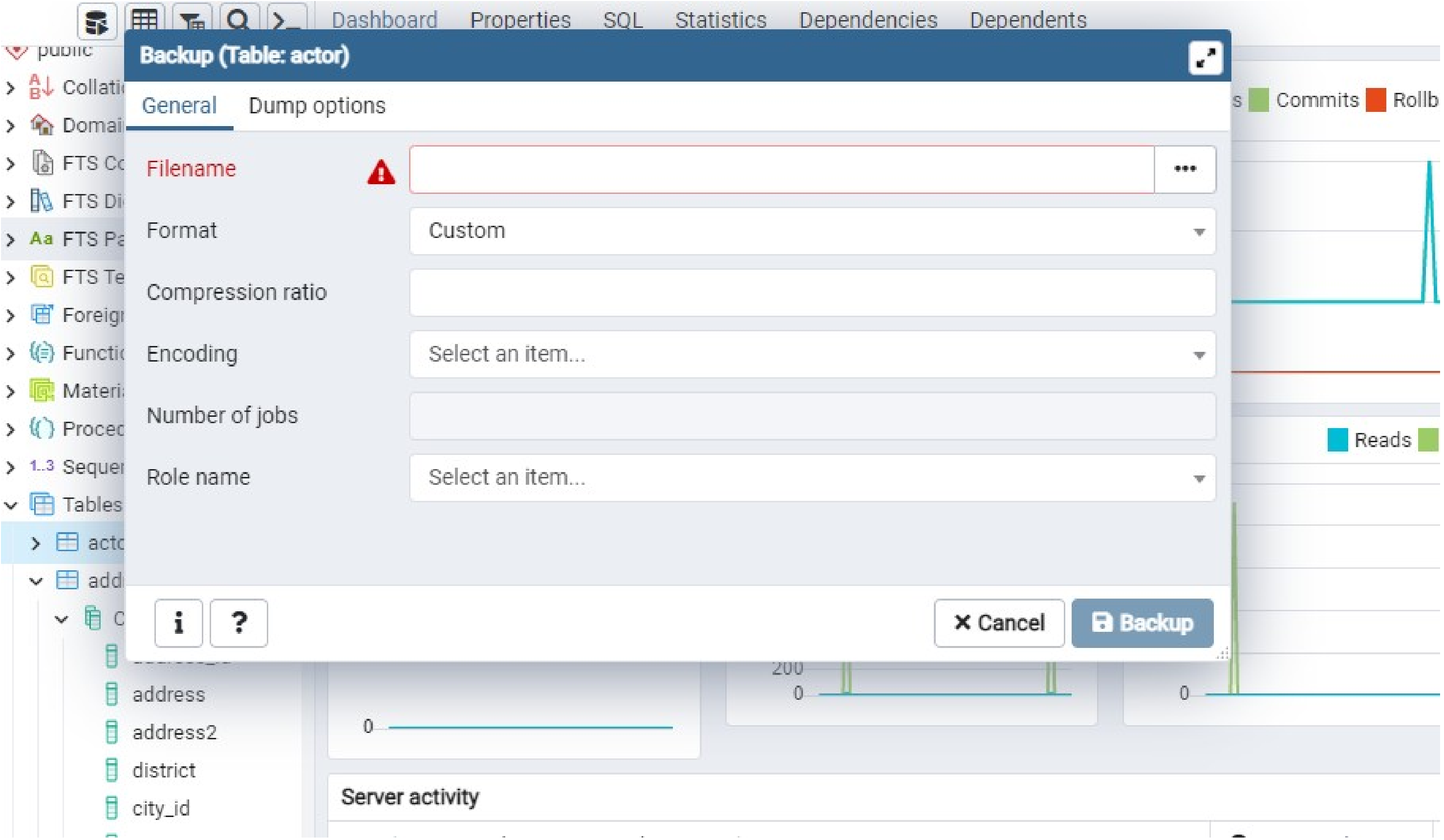
□	Object Type	Schema	Name
□	Function	public	_group_concat(text, text)
□	Function	public	film_in_stock(p_film_id integer, p_store_id i...
□	Function	public	film_not_in_stock(p_film_id integer, p_store...
□	Function	public	get_customer_balance(p_customer_id inte...
□	Function	public	inventory_held_by_customer(p_inventory_i...
□	Function	public	inventory_in_stock(p_inventory_id integer)
□	Function	public	last_day(timestamp with time zone)
□	Function	public	rewards_report(min_monthly_purchases in...
□	Trigger Function	public	last_updated

Please select any database object.

Backups

- Tools menu:
  - Backup
  - Restore
  - Import/export
  - Maintenance
- It is also possible to save:
  - Queries
  - Results (data)

Backups



Backup (Table: actor)

Filename:

Format: Custom

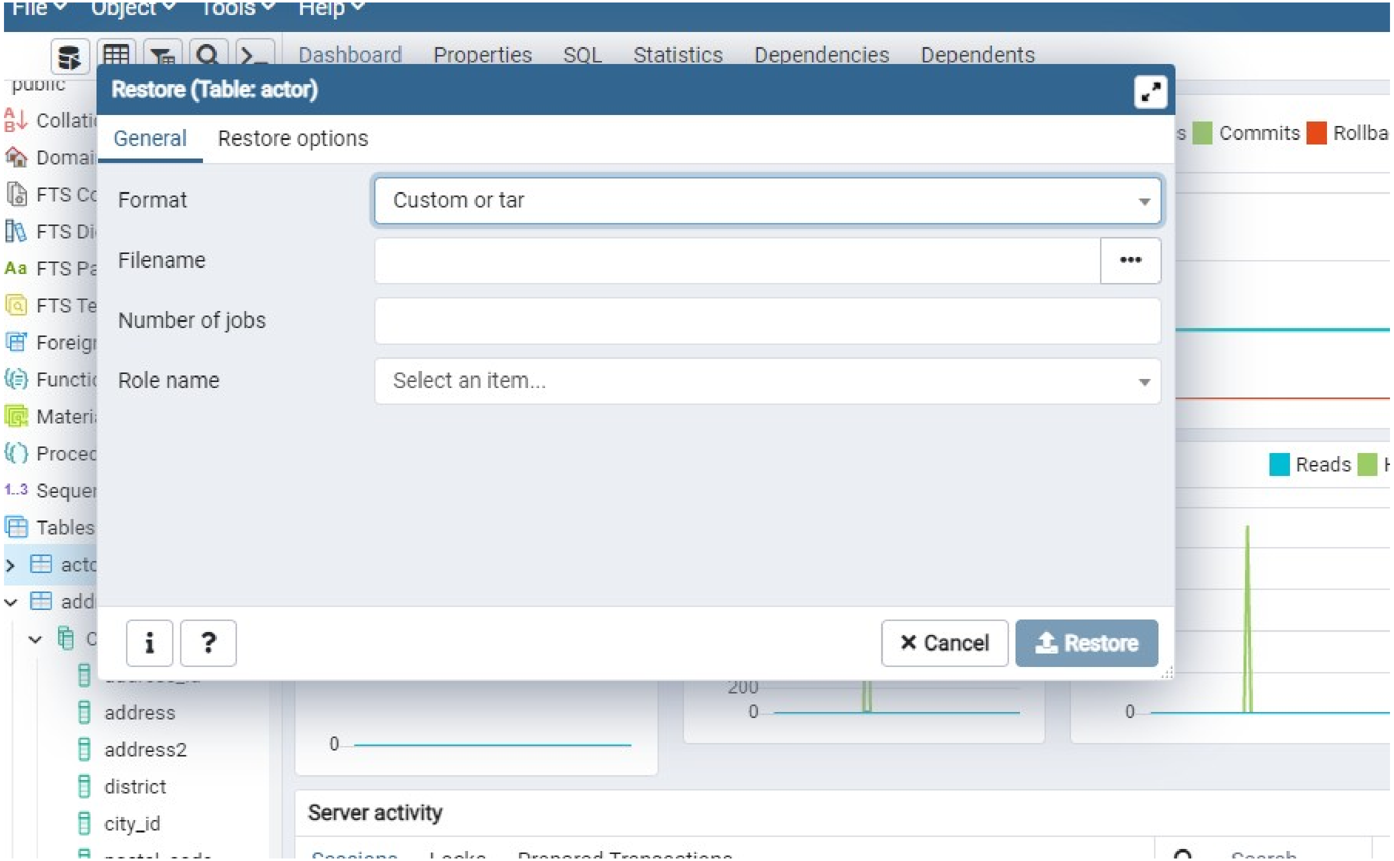
Compression ratio:

Encoding: Select an item...

Number of jobs:

Role name: Select an item...

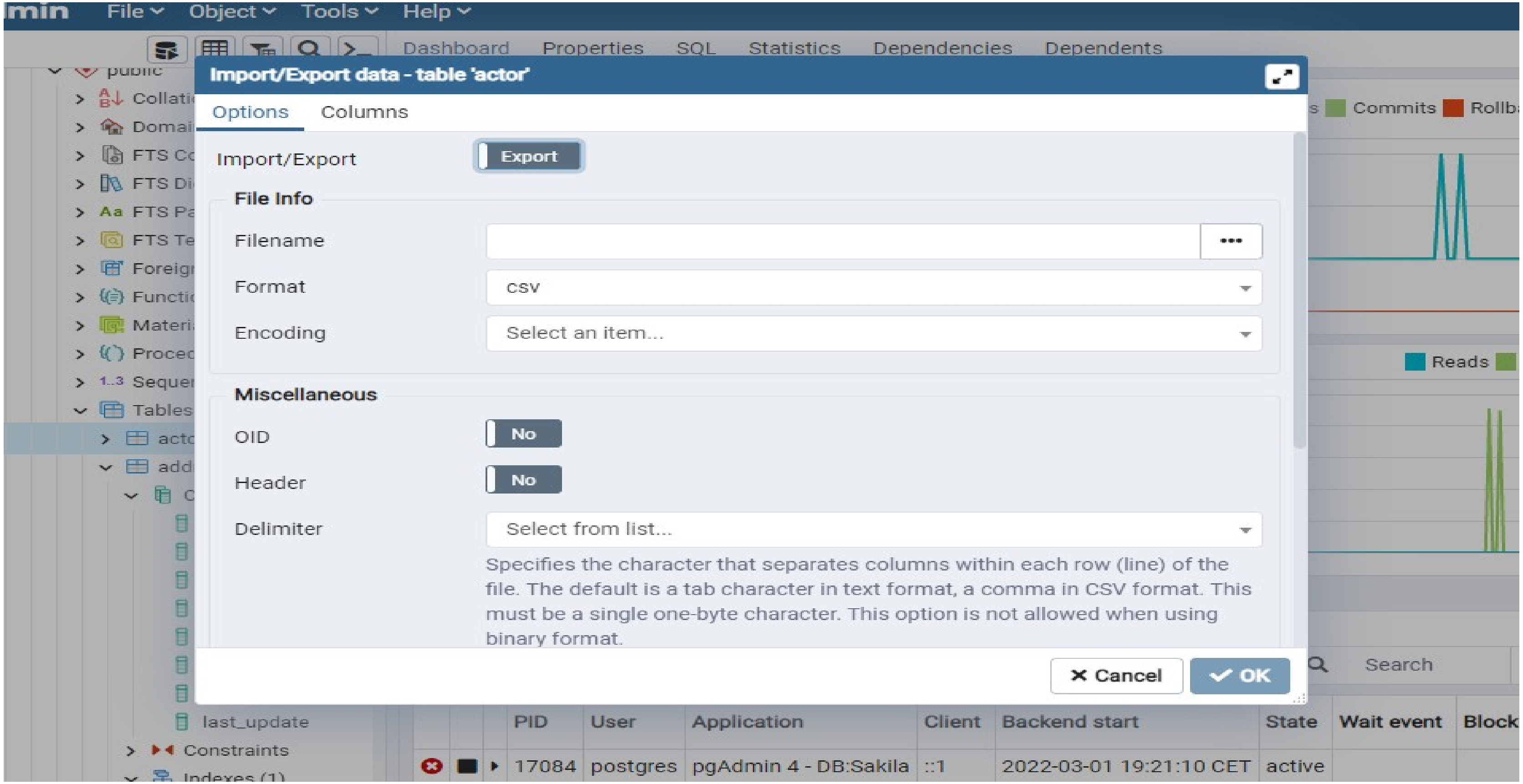
Backups



Restore (Table: actor)

Format	Custom or tar
Filename	
Number of jobs	
Role name	Select an item...

Backups



Import/Export data - table 'actor'

Import/Export: Export

File Info

Filename:

Format: csv

Encoding: Select an item...

Miscellaneous

OID: No

Header: No

Delimiter: Select from list...

Specifies the character that separates columns within each row (line) of the file. The default is a tab character in text format, a comma in CSV format. This must be a single one-byte character. This option is not allowed when using binary format.

PID	User	Application	Client	Backend start	State	Wait event	Block
17084	postgres	pgAdmin 4 - DB:Sakila	::1	2022-03-01 19:21:10 CET	active		

Indices

- Example: Find a person's phone
  - With data sorted (alphabetically)
  - No sorted data
- Solution → define an index that allows traversing the table quickly
- An index is nothing more than a binary tree that will allow traversing and searching to be efficient

Indices

- Disadvantages of indices:
  - Additional storage space required
  - Efficiency limits → above a certain threshold, having more indexes does not speed up searches
- Criteria for creating or finding index candidates:
  - Highly requested data
  - Data that can be sorted

Indices

- Syntax:

  ```
  CREATE INDEX <index name>
  ON <table> (
  column [ASC | DESC] [NULLS {FIRST | LAST}], ...
  );
  ```

- Indexes can be created on one column or on several (multi-column index)

Indices

- Example:

  ```
  CREATE INDEX idx_conductor_matricula
  ON driver (number_plate);
  ```

- … and then:

  ```
  EXPLAIN SELECT *
  FROM driver
  WHERE number_plate = 'AAA111';
  ```

Indices

- We can check the indices in an easy way by using pgAdmin
- Keys (primary, foreign, unique) always implement a (non-explicit) index by default
- How to delete an index:

```
DROP INDEX idx_conductor_matricula;
```

Indices

- Exercise:
  - Check the indexes on the clients in Sakila database
  - Check the indexes on the addresses
  - Create a new index that allows you to find more efficiently:
    - Data of a client such as his email (searching by name for example)
    - Data from an address such as your phone (searching by postal code or district, for example)
  - Compare the results after adding the indices (using the EXPLAIN statement)