```
SELECT
id,
 (SELECT name FROM users WHERE users.id=private videos orders.customer id) AS
customer_name,
 (SELECT name FROM users WHERE users.id=private_videos_orders.owner_id) AS
owner name,
  (SELECT
     (SELECT name FROM payment_systems WHERE
payment systems.id=payments.payment system id)
  FROM payments WHERE payments.order_id=private_videos_orders.id) AS
payment_system,
      order_date
FROM private videos orders
ORDER BY order date DESC LIMIT 20;
SELECT order_id,
 payment_systems.name AS payment_system,
 payment statuses.name AS status,
 private video id,
 size
FROM payments
 JOIN private videos orders
  ON payments.order_id = private_videos_orders.id
 JOIN payment_statuses
      ON payment statuses.id = payments.status
 JOIN payment systems
      ON payment systems.id = payments.payment system id
 LEFT JOIN videos
  ON videos.id = private videos orders.private video id
```

ORDER BY size DESC
LIMIT 10;
6
SELECT order_id,
status,
price,
private_video_id
FROM payments
JOIN private_videos_orders
ON order_id = private_videos_orders.id
SELECT videos.id,
title,
category_public AS categroy,
name
FROM videos
JOIN users
ON author_id = users.id
JOIN categories
on category = categories.id
LIMIT 10;

CREATE OR REPLACE VIEW video_view AS SELECT videos.id, videos.size, users.name

```
FROM videos
  LEFT JOIN users
   ON videos.author_id = users.id
 WHERE videos.size > 23;
SELECT * FROM video_view LIMIT 5;
CREATE OR REPLACE VIEW video_orders AS
SELECT private videos orders.id, private videos orders.order date, payments.status,
payments.price
 FROM private_videos_orders
  LEFT JOIN payments
   ON private videos orders.id = payments.order id
 WHERE private videos orders.order date > (current timestamp - interval '3 month');
SELECT * FROM video_orders;
CREATE FUNCTION video customers(user id INTEGER)
RETURNS INTEGER AS
$$
SELECT customer_id
FROM private videos orders
WHERE owner id = user id
GROUP BY customer_id
ORDER BY count(*) DESC
LIMIT 5;
$$
LANGUAGE sql;
```

```
SELECT video_customers(55);
CREATE OR REPLACE FUNCTION video_cheak_before_delete_trigger()
RETURNS TRIGGER AS
$$
BEGIN
 IF OLD.category = 1 THEN
 RAISE EXCEPTION 'Active orders are found. Video can not be removed.';
  END IF;
 RETURN NEW;
END;
$$
LANGUAGE plpgsql;
DROP TRIGGER IF EXISTS video_cheak_before_delete ON videos;
CREATE TRIGGER video cheak before delete
BEFORE DELETE ON videos
FOR EACH ROW
EXECUTE FUNCTION video_cheak_before_delete_trigger();
SELECT * FROM videos
DELETE FROM videos
WHERE id = 99;
```

ERROR: Active orders are found. Video can not be removed.

10-----

EXPLAIN ANALYZE SELECT order_id,

status,

price,

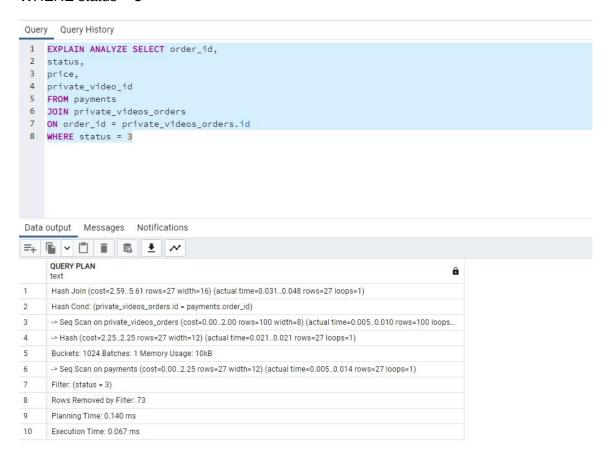
private_video_id

FROM payments

JOIN private_videos_orders

ON order_id = private_videos_orders.id

WHERE status = 3



Индекс

CREATE INDEX payments_status_fk ON payments (status);

CREATE INDEX payments_order_id_fk ON payments (order_id);