

### 1. Запустить контейнеры











```
docker-compose -f ./shards-compose.yml up --scale worker=2 -d
```

### 2. Создадим распределенную (шардированную) таблицу:

```
SELECT create_distributed_table('dialogs', 'author_id');
```

### 3. Посмотрим, видит ли координатор шарды:

```
SELECT shard_count FROM citus_tables WHERE table_name::text = 'dialogs';  
SELECT master_get_active_worker_nodes();
```

Data Output		Messages	Notifications
        			
	master_get_active_worker_nodes record 		
1	(docker-worker-1,5432)		
2	(docker-worker-2,5432)		

### 4. Наполним данными

```
insert into dialogs(author_id, user_id, message)  
select  
  i,  
  i+1,  
  md5(random())::text  
from generate_series(1, 1000000) as i;
```

### 5. Посмотрим план запроса

```
explain select * from dialogs limit 10;
```

Data Output		Messages	Notifications
<div><div><div><div>≡+</div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div></div></div></div>			
	QUERY PLAN		
	text		
1	Limit (cost=0.00..0.00 rows=10 width=56)		
2	-> Custom Scan (Citus Adaptive) (cost=0.00..0.00 rows=100000 width=56)		
3	Task Count: 32		
4	Tasks Shown: One of 32		
5	-> Task		
6	Node: host=docker-worker-1 port=5432 dbname=postgres		
7	-> Limit (cost=0.00..0.21 rows=10 width=57)		
8	-> Seq Scan on dialogs_102008 dialogs (cost=0.00..674.50 rows=31550 width=...		

### 6. Добавить шардов

```
docker-compose -f ./shards-compose.yml up --scale worker=5 -d
```










## 7. Посмотрим, видит ли координатор новые шарды:

```
SELECT master_get_active_worker_nodes();
```

Data Output		Messages	Notifications
	<div> <div>+</div> <div></div> <div>▼</div> <div></div> <div>▼</div> <div></div> <div></div> <div></div> <div></div> </div>		
	<b>master_get_active_worker_nodes</b> record		
1	(docker-worker-1,5432)		
2	(docker-worker-3,5432)		
3	(docker-worker-4,5432)		
4	(docker-worker-2,5432)		
5	(docker-worker-5,5432)		

## 8. Проверим, на каких узлах лежат сейчас данные:

```
SELECT nodename, count(*)
FROM citus_shards GROUP BY nodename;
```

Data Output	Messages	Notifications			
	 	  			
	<b>nodename</b> text	<b>count</b> bigint			
1	docker-worker-1	16			
2	docker-worker-2	16			

9. Установим wal\_level = logical чтобы узлы могли переносить данные:

```
alter system set wal_level = logical;  
SELECT run_command_on_workers('alter system set wal_level = logical');
```

10. **Перезапускаем все узлы в кластере, чтобы применить изменения wal\_level.**

В pg\_hba.conf шардов установить host all all all trust  
docker-compose -f ./shards-compose.yml restart

### 11. Проверим, что wal\_level изменился:

```
show wal_level;
```

Data Output		Messages
	wal_level text	
1	logical	

12. В таблице нет первичного ключа, п.э. устанавливаем режим шардирования

```
SELECT rebalance_table_shards(  
  'dialogs',  
  shard_transfer_mode => 'force_logical'  
);
```

13. Запустим ребалансировку:

```
SELECT citus_rebalance_start();
```

14. Проверяем, что данные равномерно распределились по шардам:

```
SELECT nodename, count(*)  
FROM citus_shards GROUP BY nodename;
```

Data Output			Messages	Notifications
	nodename text	count bigint		
1	docker-worker-1	7		
2	docker-worker-2	7		
3	docker-worker-3	6		
4	docker-worker-4	6		
5	docker-worker-5	6		