

# Введение в PostgreSQL в облаках: технологии для современных DBA

PostgreSQL Advanced



Проверить, идет ли  
запись

# Меня хорошо видно & слышно?





## Виктор Коробков

ООО «ИТ ИКС5 Технологии»

- 20+ опыт в базах данных
- Опыт преподавания off-line и on-line
- В Otus уже 7 лет
- Руководитель курсов: PostgreSQL для администраторов баз данных и разработчиков, PostgreSQL Advanced, Computer Science, Программист С.



@Korobkov\_Viktor



# Правила вебинара



Активно  
участвуем



Задаем вопрос  
в чат или голосом



Вопросы вижу в чате,  
могу ответить не сразу

# Правила вебинара



Off-topic обсуждаем в закрытом чате в Telegram:  
OTUS Postgre-Advanced-2025-12

Кто еще не присоединился:

[https://t.me/+bDFLJJwnW\\_s2M2My](https://t.me/+bDFLJJwnW_s2M2My)

# Маршрут вебинара

Знакомство

О курсе

Немного истории

Виды и выбор СУБД

Масштабирование и бекапирование

Архитектура PostgreSQL



# Цели вебинара

К концу занятия вы сможете

1. Познакомиться с преподавателями и одногруппниками
2. Рассмотреть программу курса
3. Понять, как проходит обучение в OTUS
4. Разобраться где сейчас PostgreSQL



# Знакомство

---



# Команда курса

<https://otus.ru/lessons/postgresql-cloud/>

**Алексей Железной:** Senior Data Engineer, разрабатывает и проектирует пайплайны для обработки, хранения и визуализации данных.

**Михаил Ржевский:** сертифицированный разработчик Dynamics AX и Dynamics CRM, Microsoft Certified Trainer.

**Николай Лапшин:** архитектор и Team Lead, разрабатывает решения в сфере Digital Advertising, программирует на JavaScript, TypeScript, Golang.

**Владимир Дроздецкий:** DevOps TeamLead at Magnit-Online, активно развивает системы мониторинга и логирования: Elasticsearch, Grafana Loki, Prometheus, Alertmanager, Grafana, etc.

**Дмитрий Золотов:** FullStack разработчик (Flutter/Python/Kotlin/GoLand/C++), DevOps, системный администратор, приглашенный преподаватель в ИТМО.



# Расскажите о себе

Напишите, пожалуйста, в чат или скажите голосом



Как вас зовут?



Какой опыт в IT?



Какие ожидания от курса?



Заполните информацию в разделе «О себе» в личном кабинете

**О курсе**

---

# Карта курса



СТАРТ

Введение

PostgreSQL в облаках и  
автоматизация процессов

PostgreSQL и большие данные

Проектная работа

ФИНИШ



# Практика на курсе

## 1. Облака:

Yandex <https://cloud.yandex.ru/docs/cli/quickstart>

VKCloud <https://mcs.mail.ru/>

SberCloud <https://sbercloud.ru/ru>

Selectel <https://selectel.ru/services/cloud/servers/>

## 2. Docker: <https://www.docker.com>

## 3. Виртуальная машина: <https://www.virtualbox.org/wiki/Downloads>

## 4. Windows with WSL: <https://learn.microsoft.com/en-us/windows/wsl/install>

## 5. On-premise: <https://www.postgresql.org/download>



# Как сдавать ДЗ?

Инструкции как подключиться к вебинару и как сдавать ДЗ есть в ЛК.

ДЗ сдаем в формате маркдауна в github.

Для проверки ДЗ копируем ссылку на github в чат с преподавателем в ЛК. Один репозиторий на весь курс (otus-PostgresAdvanced-2025-12-<ваше имя>).

Вопросы по ДЗ в чат ЛК, общие вопросы по ДЗ (и не только) лучше в Telegram.

Если у преподавателя есть вопросы, ДЗ отправляется на доработку. Среднее время проверки ДЗ 2-3 суток, но может быть и дольше. :))

Дедлайнов в течении курса нет. Дедлайн – окончание курса 29.06.2026.

**В сертификате будет указано количество сданных ДЗ.**



# GIT

Напишите в чат ваш уровень знакомства с GIT / GitHub:

0 – не знаком совсем;

1 – слышал, читал, даже пробовал, но в работе не использую;

2 – использую каждый день;

3 – гуру ))



# GIT

GUI:

<https://desktop.github.com/>

<https://git-scm.com/>

<https://tortoisegit.org/>

Материалы :

<https://git-scm.com/book/ru/v2>

<http://www-cs-students.stanford.edu/~blynn/gitmagic/intl/ru/>

<https://guides.github.com/activities/hello-world/>

Интерактивные туры:

<https://githowto.com/ru>

[https://learngitbranching.js.org/?locale=ru\\_RU](https://learngitbranching.js.org/?locale=ru_RU)

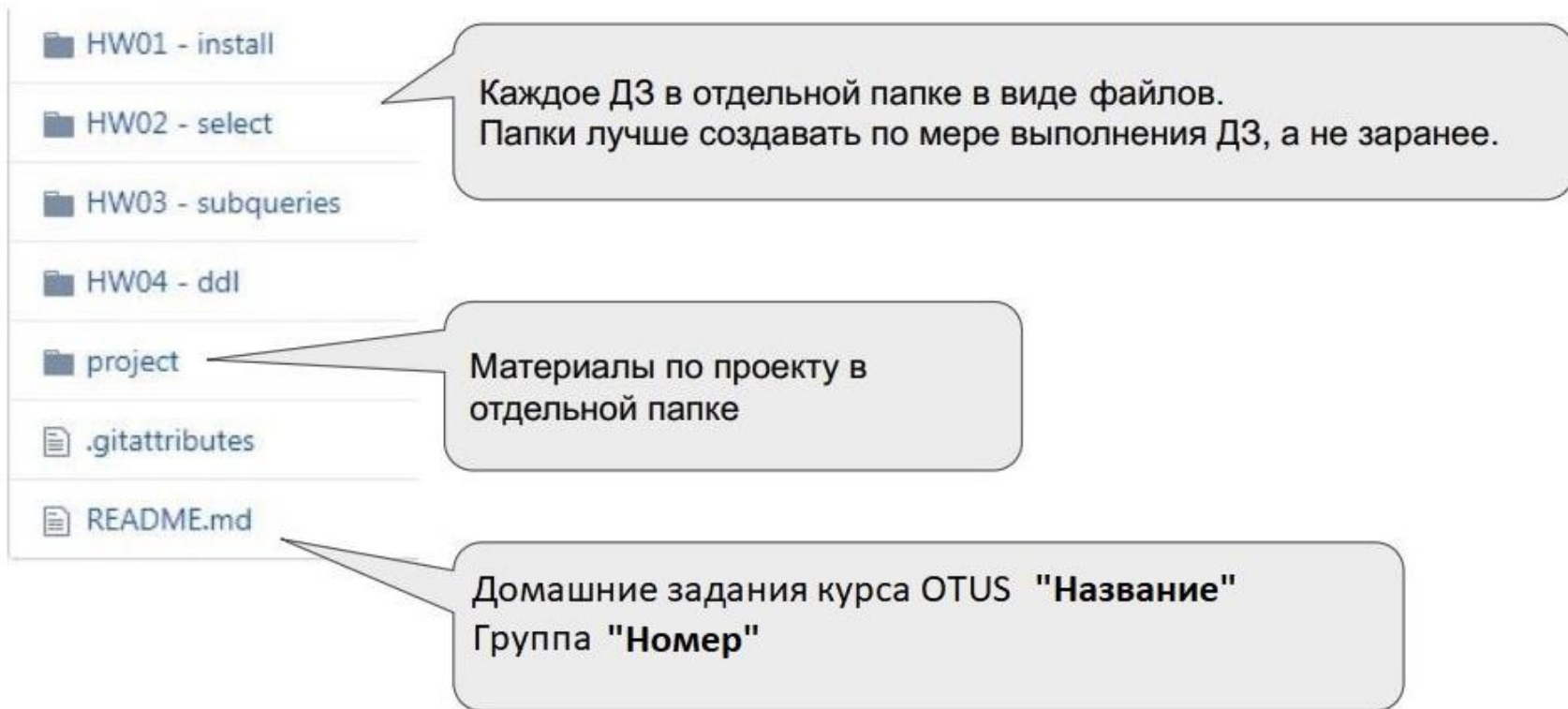
Видеокурс от OTUS

<https://otus.ru/online/git>





# Пример структуры репозитория для ДЗ



# Результат от курса

## 1) Что получите от курса:

- a) научитесь эффективно настраивать PostgreSQL
- b) получите навыки работы с PostgreSQL в облаках
- c) научитесь разворачивать отказоустойчивые кластера
- d) узнаете как в PostgreSQL работать с большими данными

## 2) Что создадите в качестве проекта

- a) высоконагруженные отказоустойчивые кластеры PostgreSQL на базе Patroni
- b) оптимизируете настройки и структуру объектов для запросов на больших объемах данных
- c) выполните миграцию данных в PostgreSQL



**Список литературы и ПО — в учебном чате**

**Доп. материалы — в личном кабинете, раздел “материалы”**



# Как будем учиться?



## Вебинары

**Вторник, пятница, 20:00**

Запись и материалы выкладывают, как правило, на следующий день после вебинара.



## Домашние задания

**Всего 13 штук + проект.**

Дедлайна нет, кроме окончания курса :)  
Типовой срок проверки: 2-3 дня.



## Чат в Telegram

**Задавайте вопросы, обменивайтесь инсайтами.**

# Лайфхаки



Если не успеваете охватить весь материал, сделайте упор на темы, которые для вас наиболее актуальны



Старайтесь применять полученные знания на практике, найдите стажировку, если у вас есть такая возможность



Смело задавайте вопросы, так материал лучше усваивается



Регулярно выполняйте ДЗ, т.к. наверстать пропуски тяжело



**Больше полезной информации  
в нашем курсе **OTUS Heroes**  
или в [курсе о том, как учатся взрослые](#)**

# Немного о мотивации

Приготовьтесь к волнообразной реакции от «ух ты как интересно» до «куда я попал?»

В такие моменты вспоминайте про цели обучения, которые вас мотивируют.

Это нормально :)



А если вы еще не определились с целью, загляните в [наш гайд по целеполаганию](#)



# ОТЗЫВЫ

Вебинары во многом адаптивны.

Мы наблюдаем за вашим обсуждением в Telegram,  
берем в работу идеи из ваших отзывов на занятия  
и домашние задания и становимся лучше!



**Пожалуйста, заполняйте опросы после занятий и  
ДЗ, мы читаем все ваши сообщения :)**





Ссылки на **опросы по занятиям** преподаватели присылают в чат в конце вебинара



Ссылки на **опросы по ДЗ** приходят в чат проверки ДЗ после того, как ваша работа зачтена

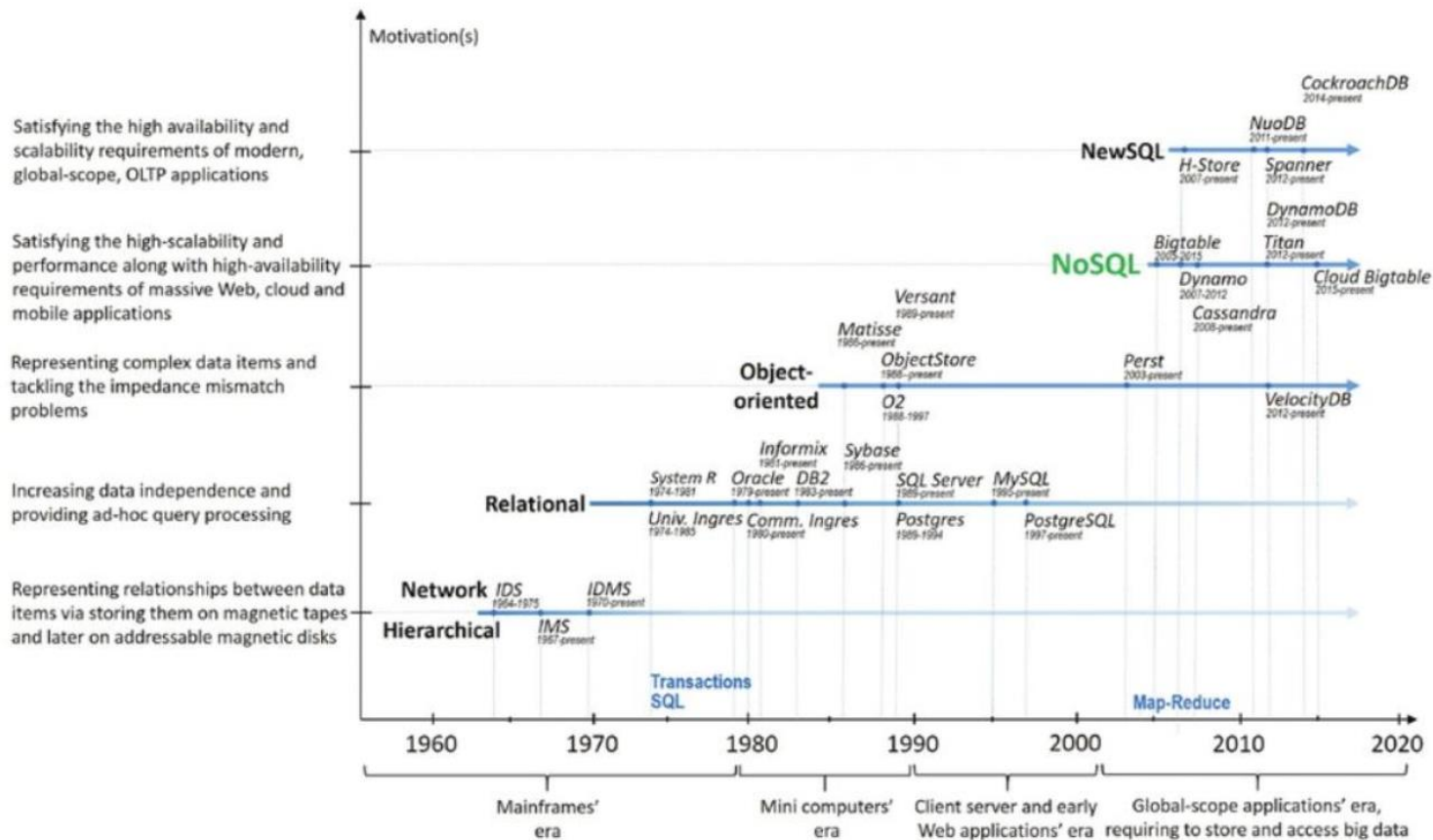
# Вопросы ?

---

# Немного истории

---

# СУБД



# Что требовалось **вчера** от DBA ?

1. Установить СУБД
2. Настроить СУБД
3. Создать:
  - ✓ табличные пространства;
  - ✓ схемы;
  - ✓ пользователей;
  - ✓ ...
4. Настроить резервное копирование
5. Настроить мониторинг
6. Отслеживать неоптимальные запросы

# Что требуется **сейчас** от DBA ?

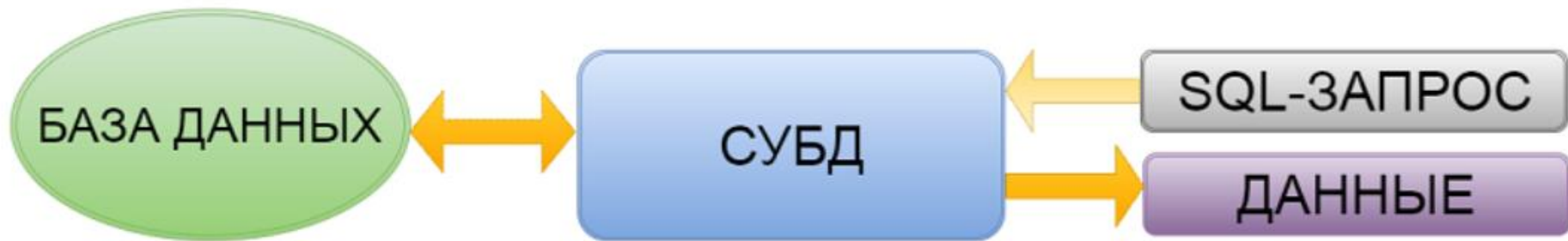
1. Выбор оптимальной СУБД или другой технологии хранения и обработки данных
2. Масштабирование нагрузки (вертикальная, горизонтальная), следовательно, репликация и шардирование, балансировка нагрузки
3. docker, k8s
4. OLTP, OLAP, оптимизации, BI, витрины данных
5. Знание инфраструктуры
6. Знание ОС
7. Миграция данных между разными СУБД
8. Переход в облака – **какие знаете ?**
9. ...



# Виды СУБД

---

# SQL СУБД



- способ взаимодействия - SQL
- внутри базы - таблицы
- транзакции, ACID
- не масштабируемы (или плохо масштабируемые)



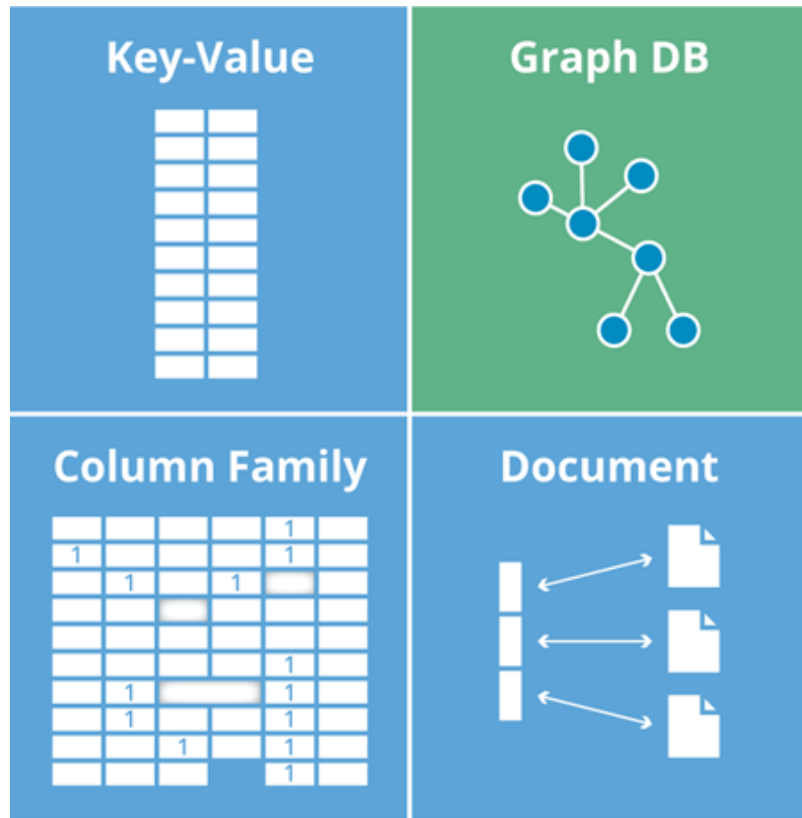
# noSQL СУБД

## NoSQL

- способ взаимодействия - поиск по ключу/ам
- внутри базы - key-value, или json, или граф, или...
- нет ACID
- scale

## Например

- MongoDB
- Cassandra
- Elasticsearch
- Google Datastore
- Amazon DynamoDB



# newSQL СУБД

## NewSQL

- способ взаимодействия - SQL
- внутри базы – таблицы, JSON
- ACID
- scale

## Например

- Amazon Aurora (MySQL dialect)
- Google Spanner (MySQL dialect)
- Galera for MySQL
- CockroachDB (PostgreSQL compatible)

# Где мы сейчас ?

в облаке (около 70% решений) или на пути к нему:

- выбор технологий СУБД, даже в формате on premises - огромный
- к этому добавляется то что есть в облаке в виде DBaaS (Database-as-a-service)

ограничения:

- навыки
- цена
- ряд ограничений в случае DBaaS (на примере su)



# Выбор СУБД

---

# OLTP vs OLAP

OLTP (OnLine Transaction Processing) системы

- оперирует текущими данными;
- большой процент операций записи;
- требуется высокая скорость обработки транзакция (ms);
- много конкурентных сессий;
- требуется консистентность данных -> (нормализация, блокировки).

Требования:

- время выполнения транзакция;
- конкурентная нагрузка;
- минимальное время простоя;
- Бэкап!!!



# OLTP vs OLAP

OLAP (OnLine Analytical Processing) системы

- получает данные посредством ETL (Extract Transform Load) из многих источников;
- часто денормализована;
- длинные транзакции;
- используется для аналитики и систем принятия решений;
- требует много индексов;
- оперирует данными за несколько лет;
- скорость обработки запроса может измеряться часами.



# Примеры

## OLTP

- Магазин, биржа, биллинг в телекоме, банковская система
- Витрины товаров, каталоги
- БД для микросервисов
- Системы управления конфигурациями

## DWH

- аналитика вышеперечисленных систем
- системы логирования
- ...



# Характеристики при выборе СУБД

- объем данных, оценка роста;
- размер транзакций (длительность, объем данных, кол-во одновременных транзакций);
- количество одновременных соединений/сессий;
- соотношение операций чтения/записи;
- масштабирование;
- партиционирование / шардирование;
- репликация;
- оценка производительности;
- организация резервного копирования;
- обеспечения высокой доступности;
- ...





# Масштабирование

---

# Виды масштабирования

**вертикальное масштабирование** : означает добавление дополнительных ресурсов к одному узлу в системе



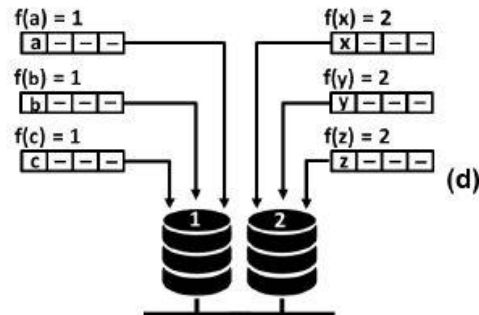
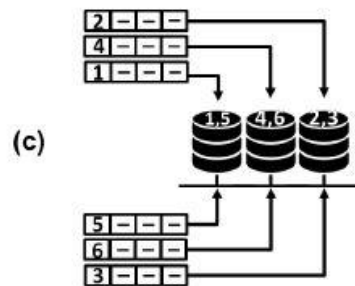
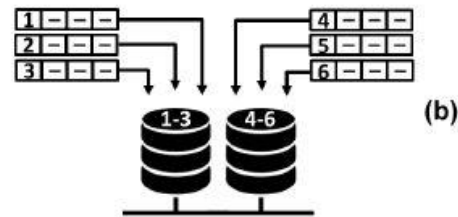
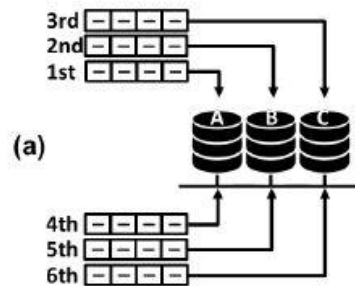
**горизонтальное масштабирование**: означает добавление дополнительных узлов в систему



# Партиционирование (partitioning)

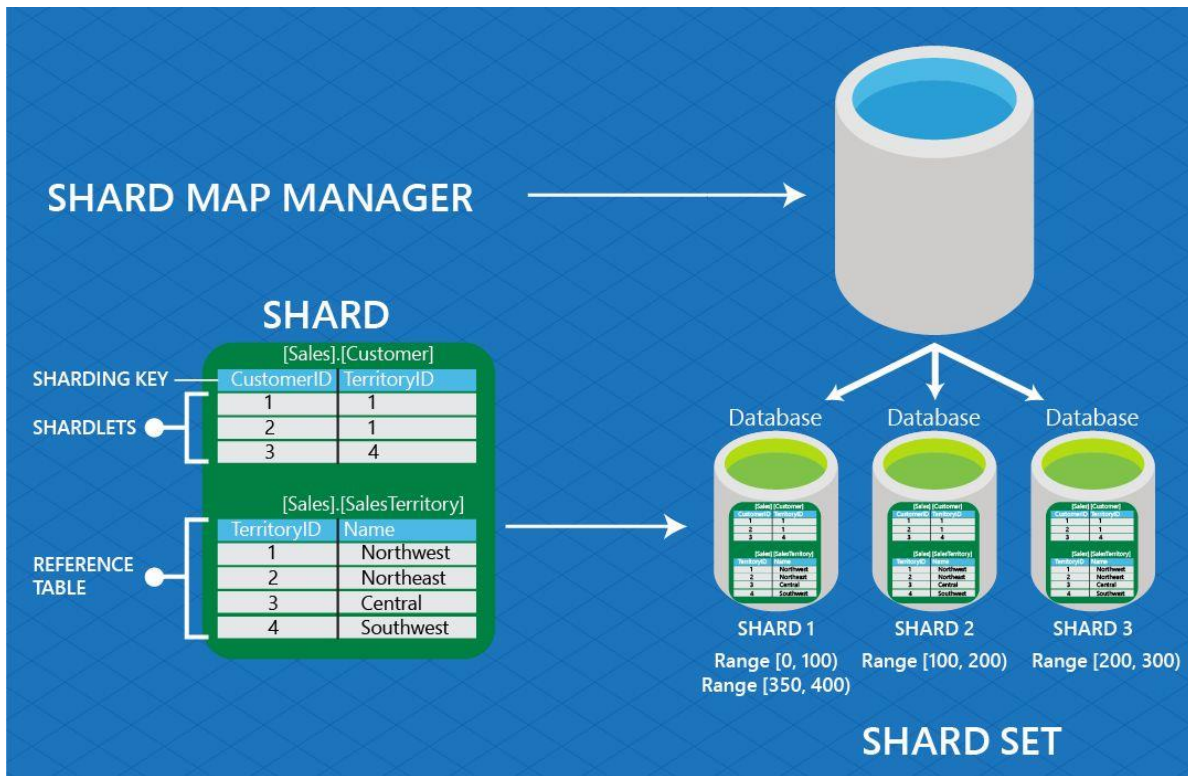
Секционирование или партиционирование – разбиение большой таблицы на секции, по ключу секционирования.

- a) Round-robin (круговая)
- b) Range (по диапазону)
- c) List (по списку)
- d) Hash (по хеш-ключу)



# Шардирование (sharding)

Шардирование (развитие партиционирования) – это разбиение данных на группы и хранение каждой группы на отдельном сервере (шарде).



# Резервное копирование

---

# Бекапы

## ПОЛНАЯ КОПИЯ



## ДИФФЕРЕНЦИАЛЬНАЯ КОПИЯ



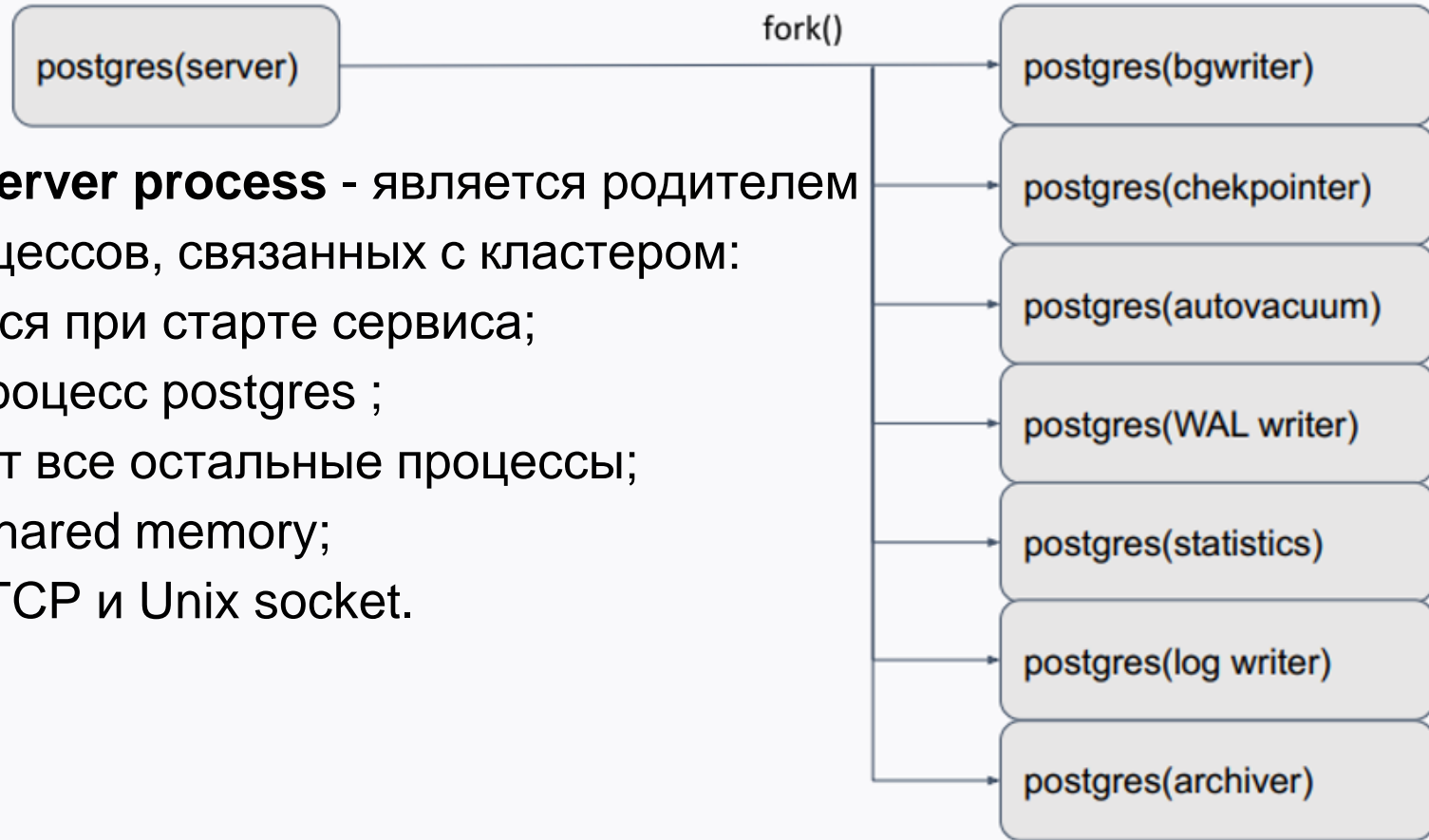
## ИНКРЕМЕНТНАЯ КОПИЯ



# Архитектура PostgreSQL

---

## PostgreSQL server

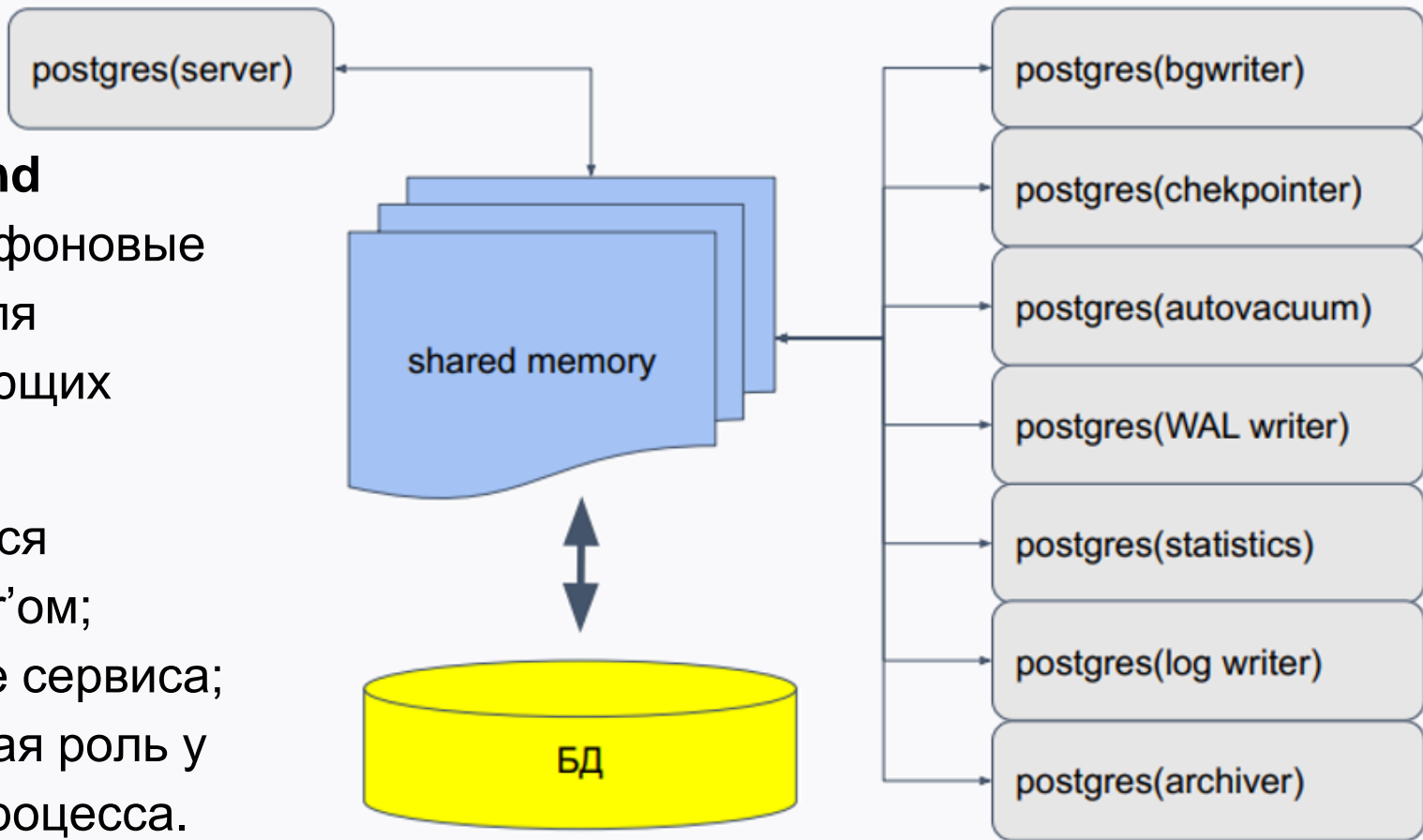


**1. postgres server process** - является родителем для всех процессов, связанных с кластером:

- запускается при старте сервиса;
- первый процесс postgres ;
- порождает все остальные процессы;
- создает shared memory;
- слушает TCP и Unix socket.



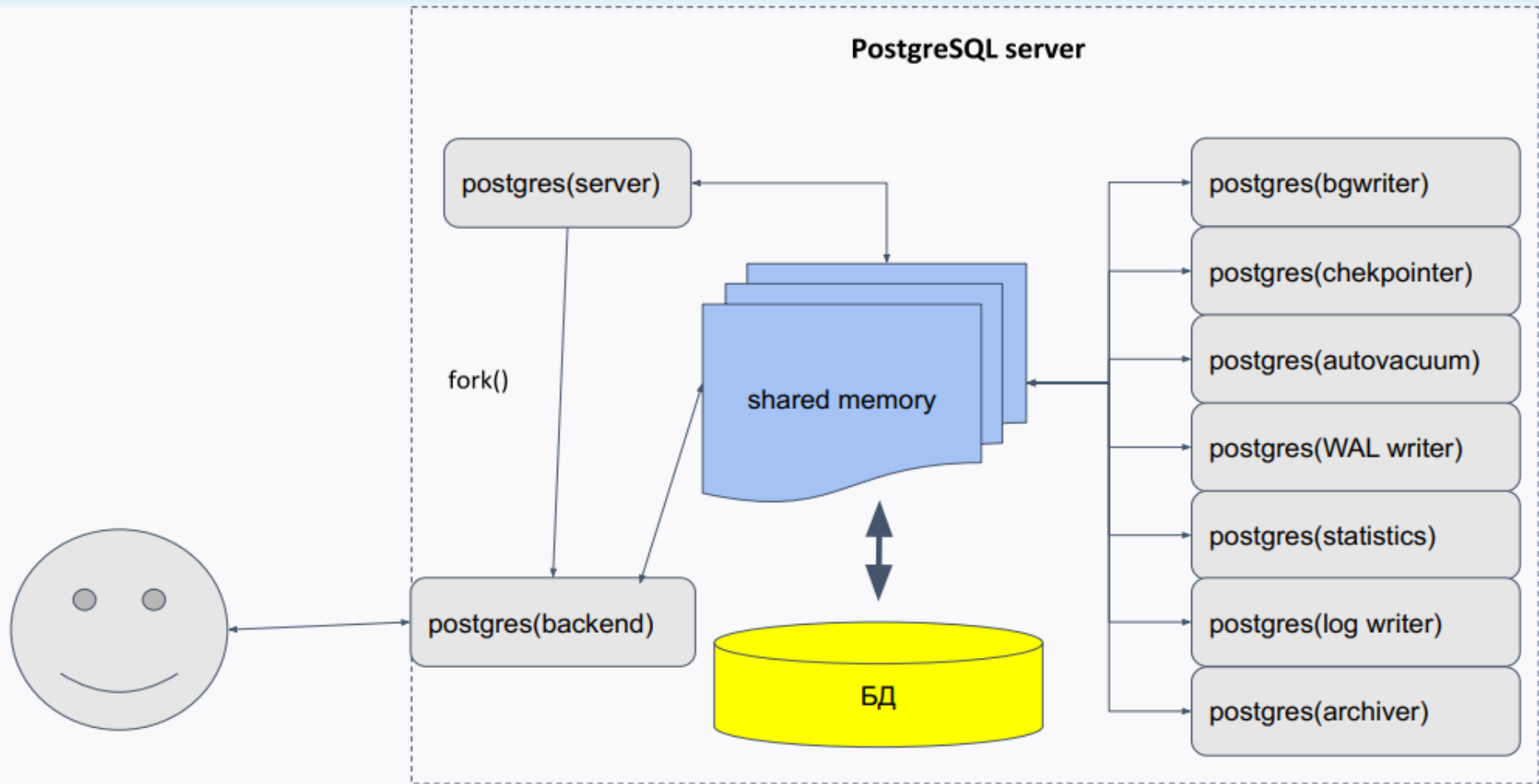
## PostgreSQL server



## 2. background

**processes** - фоновые процессы, для соответствующих функций:

- запускаются `postmaster`'ом;
- при старте сервиса;
- выделенная роль у каждого процесса.



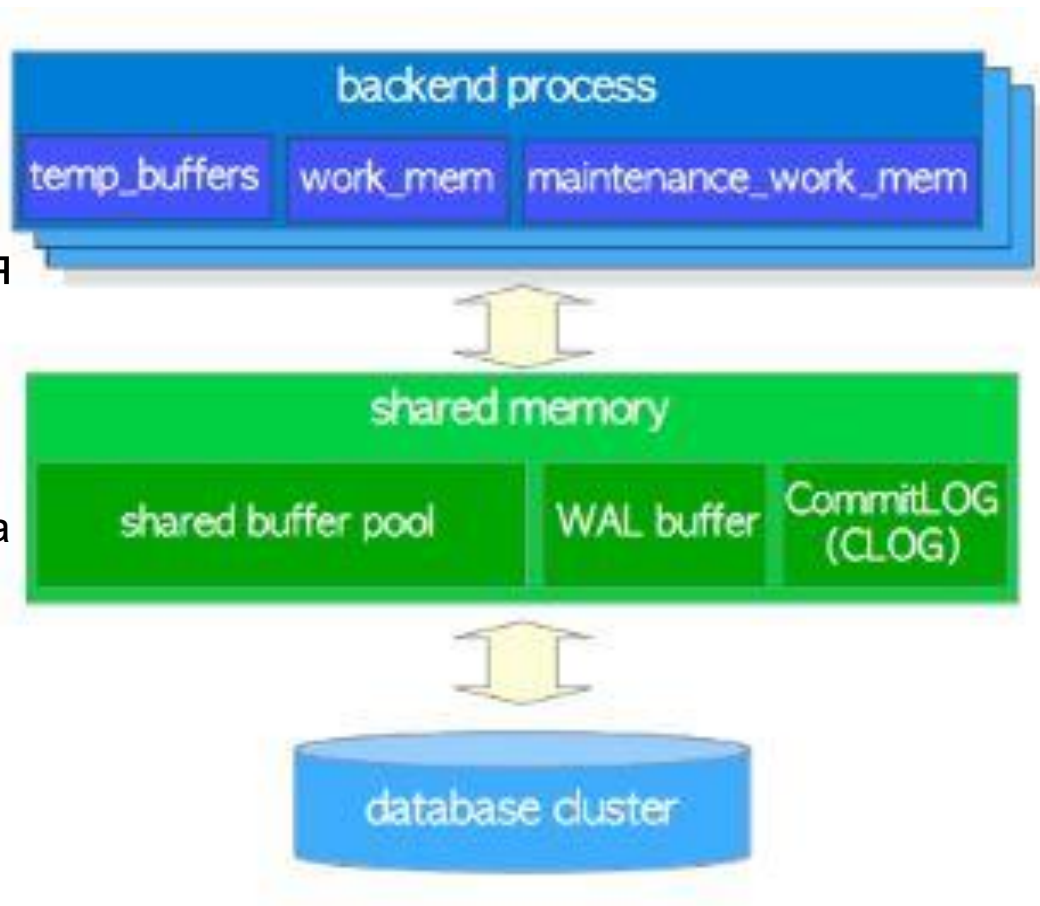
3. **backend processes** - для каждого клиентского соединения порождается выделенный серверный процесс, который обрабатывает все запросы клиента:

- запускается postmaster'ом;
- обслуживает сессию;
- работает пока сессия активна;
- максимальное количество определяется параметром `max_connections`



Local memory - выделяется каждым backend процессом для собственных нужд:

- temp\_buffers (8 Mb) - используется для хранения временных таблиц;
- work\_mem (4 Mb) - используется на этапе выполнения запроса для сортировок строк
- maintenance\_work\_mem (64 Mb) - используется служебными операциями типа VACUUM и REINDEX



# Уровни изоляции

---

# Уровни изоляции транзакций

Стандарт SQL допускает четыре уровня изоляции, которые определяются в терминах аномалий, которые допускаются при конкурентном выполнении транзакций на этом уровне:

1. «Грязное» чтение (dirty read). Транзакция T1 может читать строки измененные, но еще не зафиксированные, транзакцией T2. Отмена изменений (ROLLBACK) в T2 приведет к тому, что T1 прочитает данные, которых никогда не существовало.
2. Неповторяющееся чтение (non-repeatable read). После того, как транзакция T1 прочитала строку, транзакция T2 изменила или удалила эту строку и зафиксировала изменения (COMMIT). При повторном чтении этой же строки транзакция T1 видит, что строка изменена или удалена.
3. Фантомное чтение (phantom read). Транзакция T1 прочитала набор строк по некоторому условию. Затем транзакция T2 добавила строки, также удовлетворяющие этому условию. Если транзакция T1 повторит запрос, она получит другую выборку строк.
4. Аномалия сериализации - Результат успешной фиксации группы транзакций оказывается несогласованным при всевозможных вариантах исполнения этих транзакций по очереди.



# Уровни изоляции транзакций

	«грязное» чтение	неповторяю- щееся чтение	фантомное чтение	аномалия сериализации
Read Uncommitted	Допускается, но не в PG	да	да	да
<b>Read Committed</b>	-	да	да	да
Repeatable Read	-	-	Допускается, но не в PG	да
Serializable	-	-	-	-

Стандарт SQL допускает четыре уровня изоляции, которые определяются в терминах аномалий, которые допускаются при конкурентном выполнении транзакций на этом уровне:

[К чему может привести ослабление уровня изоляции транзакций в базах данных](#)

[PostgreSQL: 13.2. Transaction Isolation](#)



# Домашнее задание

---



1. Создать виртуальную машину с дефолтными параметрами - 1-2 ядра, 2-4Гб памяти, любой линукс (на курсе Ubuntu) в YandexCloud / VirtualBox / Docker ...
2. Добавить свой ssh ключ
3. Зайти удаленным ssh (первая сессия)
4. Поставить PostgreSQL из пакетов apt install
5. Зайти вторым ssh (вторая сессия)
6. Запустить везде psql из под пользователя postgres
7. Выключить auto commit
8. Сделать в первой сессии новую таблицу и наполнить ее данными
9. Посмотреть текущий уровень изоляции: show transaction isolation level
10. Начать новую транзакцию в обеих сессиях с дефолтным уровнем изоляции
11. В первой сессии добавить новую запись
12. Посмотреть таблицу во второй сессии. Видите ли вы новую запись и если да то почему?
13. Завершить первую транзакцию
14. Посмотреть таблицу во второй сессии. Видите ли вы новую запись и если да то почему?  
Завершить транзакцию во второй сессии.



15. Начать новые транзакции с repeatable read
16. В первой сессии добавить новую запись
17. Посмотреть таблицу во второй сессии. Видите ли вы новую запись и если да то почему?
18. Завершить первую транзакцию
19. Посмотреть таблицу во второй сессии. Видите ли вы новую запись и если да то почему?  
Завершить транзакцию во второй сессии.

ДЗ сдаем в виде миниотчета в markdown в гите



# Вопросы



если есть вопросы



если вопросов нет



**В итоге ...**

---

# Рейтинг PYPL - PopularitY of Programming Language Index

Worldwide, Dec 2025 :

Rank	Change	Database	Share	1-year trend
1		Oracle	36.36 %	+4.0 %
2		MySQL	11.71 %	-4.4 %
3		SQL Server	9.26 %	-2.1 %
4		PostgreSQL	8.61 %	+1.1 %
5		MongoDB	5.54 %	-0.4 %
6	↑↑↑↑↑ ↑↑	Supabase	4.92 %	+3.5 %
7		Redis	4.2 %	+0.4 %
8	↓↓	Microsoft Access	4.0 %	-1.6 %
9	↓	Splunk	2.67 %	-0.1 %
10	↓	SQLite	2.3 %	+0.2 %

<https://pypl.github.io/DB.html>



# Рейтинг DB-Engines

427 systems in ranking, December 2025

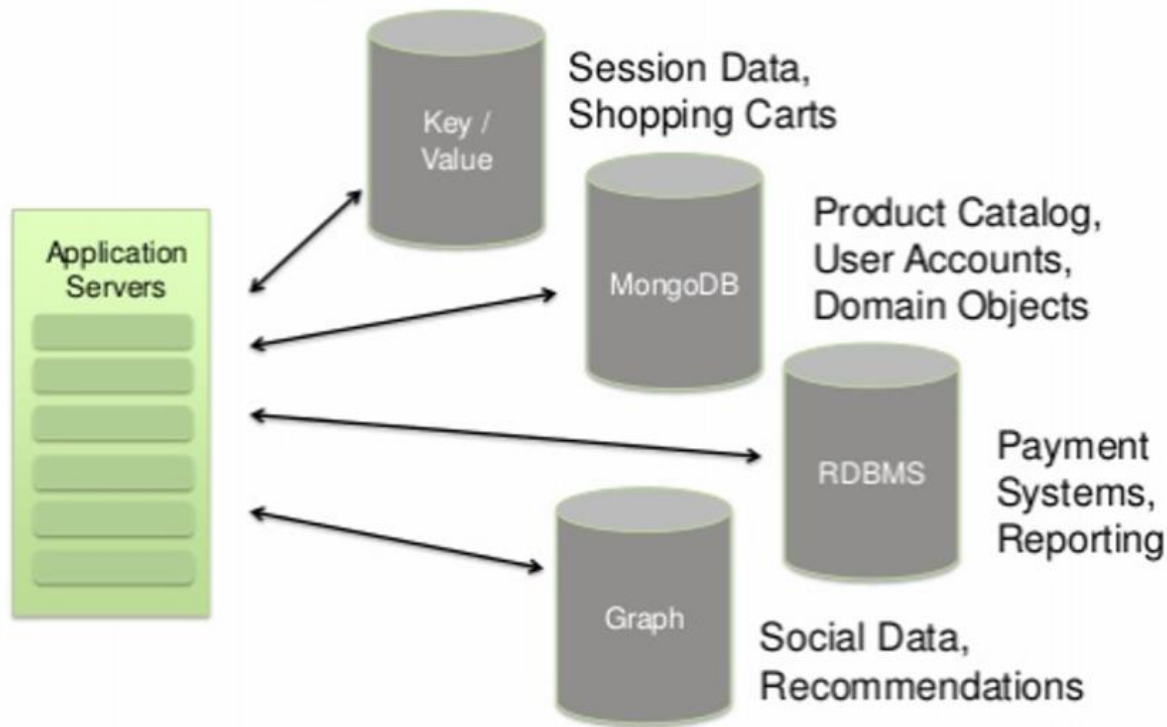
Rank			DBMS	Database Model	Score		
Dec 2025	Nov 2025	Dec 2024			Dec 2025	Nov 2025	Dec 2024
1.	1.	1.	Oracle	Relational, Multi-model ⓘ	1234.40	-5.38	-29.39
2.	2.	2.	MySQL	Relational, Multi-model ⓘ	868.49	+2.67	-135.27
3.	3.	3.	Microsoft SQL Server	Relational, Multi-model ⓘ	722.52	+3.65	-83.18
4.	4.	4.	PostgreSQL	Relational, Multi-model ⓘ	659.42	+8.06	-6.95
5.	5.	5.	MongoDB +	Multi-model ⓘ	372.27	+0.60	-28.11
6.	6.	↑ 7.	Snowflake	Relational	202.45	+4.61	+55.09
7.	7.	↓ 6.	Redis	Key-value, Multi-model ⓘ	142.48	-2.61	-7.79
8.	8.	↑ 13.	Databricks	Multi-model ⓘ	138.02	+6.51	+50.33
9.	9.	9.	IBM Db2	Relational, Multi-model ⓘ	115.75	-3.52	-7.03
10.	10.	↓ 8.	Elasticsearch	Multi-model ⓘ	109.83	-4.14	-22.49



# POLYGLOT PERSISTENCE

Реальная жизнь:

- эффективность
- сложность
- микросервисы
- большие объемы



# Итоги занятия

## Рефлексия

---





**Отправьте в чат эмодзи, который  
отражает ваше настроение  
после занятия**

# Заполните, пожалуйста, опрос о занятии

---

Мы читаем все ваши сообщения  
и берем их в работу 🖋️ ❤️

# Приглашаем на следующий вебинар: «PostgreSQL и Docker: создание образов, настройка и мониторинг» который состоится 13.01.2026



Ссылка на вебинар будет  
в ЛК за 15 минут



Материалы  
к занятию в ЛК —  
можно изучать



Обязательный материал  
обозначен красной лентой



# OTUS

С наступающим 2026 годом