ЦЕНТРАЛЬНЫЙ УНИВЕРСИТЕТ

УТВЕРЖДЕНА

Решением Ученого совета
АНО ВО «Центральный университет»
«24» июня 2025 г.
Протокол № 2

Рабочая программа дисциплины (модуля)
«SQL и базы данных для разработчиков»

Направление подготовки: *02.04.01 Математика и компьютерные науки*

Направленность (профиль) подготовки: *Backend-разработка*

Квалификация (степень) выпускника: *магистр*

Форма обучения: *очная*

Срок освоения программы: *2 года*

Год набора: *2025*

ДОКУМЕНТ ПОДПИСАН
ЭЛЕКТРОННОЙ ПОДПИСЬЮ

Сертификат: 0215B0C000D1B3FEB44A37EE631E0C27D4
Владелец: Ивашкевич Евгений Васильевич
Действителен: с 13.01.2026 до 13.04.2027

Электронный документ

Москва
2025

Содержание

1. Краткая характеристика дисциплины (модуля)

Рабочая программа дисциплины (модуля) «SQL и базы данных для разработчиков» составлена в соответствии с федеральным государственным образовательным стандартом высшего образования – магистратура по специальности 02.04.01 Математика и компьютерные науки, профиль Backend-разработка, утвержденный приказом Министерства науки и высшего образования Российской Федерации № 810 от 23.08.2017 года.

Изучение дисциплины (модуля) «SQL и базы данных для разработчиков» является ключевым для понимания того, как эффективно управлять, хранить и извлекать данные, что критически важно в современном мире, где информация играет центральную роль в принятии решений. Освоение SQL позволяет специалистам анализировать большие объемы данных, оптимизировать процессы и разрабатывать надежные решения для бизнеса и науки.

Место дисциплины (модуля) в структуре образовательной программы

Настоящая дисциплина (модуль) включена в учебный план по программе подготовки магистратуры по направлению 02.04.01 Математика и компьютерные науки, профиль Backend-разработка и входит в обязательную часть Блока 1.

Дисциплина (модуль) изучается на 1 курсе во 2 семестре.

Цель изучения дисциплины (модуля): освоение эффективного использования языка SQL и технологий баз данных для решения прикладных задач хранения, обработки и анализа данных в бизнес-среде.

Задачи изучения дисциплины (модуля):

- освоить базовый SQL и работу с данными (CRUD) в PostgreSQL;
- научиться проектировать схему данных и управлять объектами БД;
- овладеть извлечением и преобразованием данных для бизнес-задач;
- освоить аналитические запросы и оконные функции;
- научиться обеспечивать корректность и производительность: транзакции, блокировки, индексы, оптимизация и базовое администрирование PostgreSQL.

В результате освоения дисциплины (модуля) обучающийся должен:

знать:

- базовый синтаксис SQL;
- типы данных SQL и способы их преобразования;
- агрегирующие функции;
- основные функции для работы с датами, числами и текстом;
- индексы в базах данных и виды индексов;
- уровни изоляции транзакций и виды блокировок.

уметь:

- агрегировать данные;
- соединять таблицы;
- работать со множествами;
- использовать подзапросы;
- писать оконные функции;
- создавать таблицы, представления, индексы;
- оптимизировать SQL запросы.

владеть:

- навыками решения бизнес-задач с помощью SQL;
- навыками администрирования сервера PostgreSQL.

2. Перечень планируемых результатов обучения

Компетенции, формируемые в результате освоения дисциплины (модуля) при проведении учебных занятий в форме контактной работы обучающихся с педагогическими работниками Университета и в форме самостоятельной работы обучающихся:

Компетенция	Содержание компетенции	Индикатор компетенции	Перечень планируемых результатов обучения по дисциплине (модулю)
ОПК-3.	Способен самостоятельно создавать прикладные программные средства на основе современных информационных технологий и сетевых ресурсов, в том числе отечественного производства	ОПК-3.1.	Знает основные принципы программирования, архитектуры программного обеспечения и современные языки программирования, а также особенности отечественных информационных технологий и сетевых ресурсов
		ОПК-3.2.	Умеет разрабатывать прикладные программные средства, используя современные инструменты и технологии, а также интегрировать их с сетевыми ресурсами для решения конкретных задач
		ОПК-3.3.	Имеет практический опыт разработки программных средств, используемых при построении математических моделей в естественных науках
ПК-3.	Способен решать задачи профессиональной деятельности, формулировать результат, увидеть следствия полученного результата	ПК-3.1.	Знает основные принципы и методы решения задач профессиональной деятельности, а также способы формулирования и представления результатов, включая анализ последствий и их значимость в контексте проекта
		ПК-3.2.	Умеет применять математические и компьютерные методы для решения конкретных задач, формулировать четкие и обоснованные результаты, а также анализировать их последствия для дальнейших действий и решений
		ПК-3.3.	Имеет практический опыт в решении профессиональных задач, включая участие в проектах, где были получены результаты и проанализированы их следствия, что способствовало

			принятию обоснованных решений
ПК-6.	Способен разрабатывать программное обеспечение для решения прикладных задач в сфере профессиональной деятельности	ПК-6.1.	Знает основные языки программирования, методы разработки программного обеспечения, а также принципы проектирования и архитектуры программных систем, применяемых в конкретной предметной области
		ПК-6.2.	Умеет анализировать прикладные задачи, разрабатывать алгоритмы и реализовывать их в виде программного обеспечения, используя современные инструменты и технологии, а также проводить тестирование и отладку созданных решений
		ПК-6.3.	Имеет практический опыт разработки программного обеспечения в рамках реальных проектов, включая участие в командах, где были успешно реализованы решения для конкретных прикладных задач в сфере профессиональной деятельности

3. Тематический план

№ п/п	Наименование раздела дисциплины (модуля)	Трудоемкость, академические часы				ТКУ (текущий контроль успеваемости)
		Очная форма				
		Аудиторная работа		Контроль	Самостоятельная работа	
		Лекции	Семинары (практические занятия)			
1	Базовые возможности языка SQL	8	8		34	Подготовка к семинару Домашние задания
2	OLAP-базы данных и продвинутые возможности языка SQL	12	12		50	Подготовка к семинару Домашние задания Контрольная работа
3	OLTP-базы данных на примере PostgreSQL	10	10		42	Подготовка к семинару Домашние задания
	Зачет с оценкой			4		
	Итого:	***30***	***30***	***4***	***126***	
Объем дисциплины (модуля) (в ак. ч.)		***190***				
Объем дисциплины (модуля) (в зач. ед.)		***5***				

4. Содержание дисциплины (модуля)

№п/п	Наименование раздела дисциплины (модуля)	Содержание дисциплины (модуля) по темам
1	Базовые возможности языка SQL	Введение в базы данных и SQL Разработка запросов к одной таблице Агрегация в SQL Оператор JOIN. Теоретико-множественные операции
2	OLAP-базы данных и продвинутые возможности языка SQL	Подзапросы и общие табличные выражения Оконные функции Основы DDL. Модификация таблиц. Представления Оптимизация запросов. План запроса Индексы. Типы индексов Функции и процедуры в SQL
3	OLTP-базы данных на примере PostgreSQL	Транзакции. Уровни изоляции транзакций Многоверсионность. Снимки данных. Очистка и автоочистка Блокировки. Виды блокировок Буферный кеш и журнал предзаписи Масштабирование. Резервное копирование и репликация

5. Учебно-методическое обеспечение

Университет располагает полным набором лицензионного и свободно распространяемого программного обеспечения, включая продукты отечественного производства.

Каждый студент в течение всего периода обучения получает индивидуальный неограниченный доступ к электронно-библиотечной системе и электронной информационно-образовательной среде университета. Эти системы предоставляют возможность доступа к ресурсам из любой точки, где есть подключение к сети Интернет, как на территории университета, так и за его пределами.

Студентам обеспечен удаленный доступ к современным профессиональным базам данных и информационным справочным системам.

Основная литература:

1. Фиайли, К. SQL. Руководство для использования с любыми SQL СУБД : учебное пособие / К. Фиайли ; пер. с англ. А. В. Хаванова. - 2-е изд. - Москва : ДМК Пресс, 2023. - 454 с. - ISBN 978-5-89818-323-3. - Текст : электронный. - URL: https://znanium.ru/catalog/product/2102610.

2. Короткевич, Д. SQL Server. Наладка и оптимизация для профессионалов : практическое руководство / Д. Короткевич. - Санкт-Петербург : Питер, 2023. - 512 с. - (Серия «Библиотека программиста»). - ISBN 978-5-4461-2332-2. - Текст : электронный. - URL: https://znanium.com/catalog/product/2123361.

Дополнительная литература:

1. Молинаро, Э. SQL. Сборник рецептов : практическое руководство / Э. Молинаро. - 2-е изд. - Санкт-Петербург : БХВ-Петербург, 2022. - 592 с. - ISBN 978-5-9775-6759-6. - Текст : электронный. - URL: https://znanium.com/catalog/product/2123362.

2. Изучаем и используем Presto. Быстрый и надежный SQL-движок для анализа данных : практическое руководство / А. Ло Дука, Т. Михан, В. Бхаратан, Ин Су ; пер. с англ. В. И. Комарова. – Москва : Book.kz, 2024. - 184 с. – ISBN 978-6-01810-343-8. - Текст : электронный. - URL: https://znanium.ru/catalog/product/2205126.

6. Материально-техническое обеспечение

Университет располагает материально-технической базой, соответствующей действующим противопожарным правилам и нормам и обеспечивающей проведение всех видов дисциплинарной и междисциплинарной подготовки, практической и научно-исследовательской работ обучающихся, предусмотренных учебным планом.

Помещения, которые представляют собой учебные аудитории для проведения занятий лекционного типа, занятий семинарского (практического) типа, групповых и индивидуальных консультаций, текущего контроля и промежуточной аттестации, а также помещения для самостоятельной работы и помещения для хранения и профилактического обслуживания учебного оборудования. Помещения укомплектованы специализированной мебелью и техническими средствами обучения, служащими для представления учебной информации большой аудитории.

Изучение дисциплины (модуля) обеспечивается в учебных аудиториях, оснащенных:

— столами и стульями;

— компьютерной техникой;

— механическими калькуляторами;

— специализированным оборудованием, включая демонстрационное оборудование.

Помещения для самостоятельной работы обучающихся, в том числе приспособленные для использования инвалидами и лицами с ограниченными возможностями здоровья, оснащены компьютерной техникой с возможностью подключения к сети «Интернет» и

обеспечением доступа в электронную информационно-образовательную среду Университета.

Обучающимся предоставляется доступ (в том числе удаленный) к ресурсам информационно-телекоммуникационной сети «Интернет», электронным ресурсам (в том числе электронным библиотечным системам, современным профессиональным базам данных и информационным справочным системам):

№	Наименование портала (издания, курса, документа)	Ссылка
1.	Научная электронная библиотека elibrary.ru библиотека	https://elibrary.ru/defaultx.asp
2.	База данных для IT-специалистов	https://habr.com
3.	База данных ScienceDirect	https://www.sciencedirect.com
4.	Официальный сайт Министерства науки и высшего образования Российской Федерации	https://minobrnauki.gov.ru/
5.	Федеральный портал «Российское образование"	https://www.edu.ru/
6.	Информационная система "Единое окно доступа к образовательным ресурсам"	http://window.edu.ru/
7.	Единая коллекция цифровых образовательных ресурсов	http://school-collection.edu.ru/
8.	Федеральный центр информационно - образовательных ресурсов	http://fcior.edu.ru/

Перечень информационных технологий, используемых при осуществлении образовательного процесса по дисциплине (модулю), в том числе комплект лицензионного программного обеспечения, современные профессиональные базы данных и информационные справочные системы:

Наименование ПО	Производство	Лицензионное / свободно распространяемое
Операционные системы:		
Microsoft Imagine (Windows Client, Server)	зарубежное	лицензионное
Браузеры:		
Яндекс.Браузер	отечественное	свободно распространяемое
Google Chrome	зарубежное	свободно распространяемое
Офисные приложения:		
Microsoft Imagine (Visio, OneNote)	зарубежное	лицензионное
TeXstudio	зарубежное	свободно распространяемое
Adobe Acrobat Reader	зарубежное	свободно распространяемое
Программное обеспечение для планирования и учета времени:		
Toggle app	зарубежное	свободно распространяемое
Системы управления проектами:		
Microsoft Imagine (Project)	зарубежное	лицензионное
Системы управления базами данных:		
Microsoft Imagine (SQL Server)	зарубежное	лицензионное
Системы резервного копирования (backup):		
Acronis Backup Advanced for HyperV	зарубежное	лицензионное
Справочно-правовые системы:		
КонсультантПлюс: справочно-правовая система	отечественное	лицензионное
Средства антивирусной защиты:		

Kaspersky Endpoint Security для бизнеса Стандартный Russian Edition	отечественное	лицензионное
Среды разработки:		
Visual Studio Code	зарубежное	свободно распространяемое
Bash (Unix shell)	зарубежное	свободно распространяемое
Anaconda	зарубежное	свободно распространяемое
Robotic Operating System	зарубежное	свободно распространяемое
CopelliaSim	зарубежное	свободно распространяемое
Google Colaboratory	зарубежное	свободно распространяемое
Пакеты программных средств и библиотек:		
AutoPsy	зарубежное	свободно распространяемое
Interactive Disassembler (IDA)	зарубежное	свободно распространяемое
Системы управления библиографической информацией:		
Zotero	зарубежное	свободно распространяемое
Сервисы и службы:		
Bind	зарубежное	свободно распространяемое
Docker	зарубежное	свободно распространяемое

7. Методические и оценочные материалы

Методические указания для обучающихся по освоению дисциплины (модуля)

В процессе изучения дисциплины (модуля) «SQL и базы данных для разработчиков» в рамках текущего контроля успеваемости используются такие виды учебной работы, как лекции, семинары, аудиторная работа, домашние задания, контрольная работа, а также различные виды самостоятельной работы обучающихся по заданию преподавателя, направленные на развитие навыков профессиональной лексики, закрепление практических профессиональных компетенций, поощрение инициатив.

Лекция – систематическое, последовательное, монологическое изложение преподавателем учебного материала, как правило, теоретического характера.

В процессе лекций рекомендуется вести конспект лекций: кратко и схематично фиксировать основные идеи, выводы и обобщения лекции; выделять важные мысли, ключевые слова и термины. Необходимо отметить вопросы или материалы, которые вызывают затруднения, и попытаться найти ответы в рекомендованной литературе. Если разобраться в материале не удается, следует сформулировать вопрос и задать его преподавателю на консультации или во время семинарского (практического) занятия.

Семинар – это форма учебной деятельности, проводимая в учебном заведении под руководством преподавателя, где студенты активно участвуют в обсуждениях, практических заданиях и других формах взаимодействия.

Для успешной подготовки к семинару рекомендуется заранее ознакомиться с темой занятия и основными материалами, чтобы иметь возможность активно участвовать в обсуждении. Также полезно подготовить вопросы и идеи для обсуждения, что поможет глубже понять материал и продемонстрировать заинтересованность.

Аудиторная работа – активная работа студента на семинаре/практическом занятии, его ответы на вопросы преподавателя и участие в дискуссии.

Для успешного участия в семинаре/практическом занятии студентам рекомендуется заранее ознакомиться с темой обсуждения, прочитать необходимые материалы и подготовить вопросы. Важно активно слушать и вовлекаться в дискуссию, высказывая свои мнения и аргументируя их. При ответах на вопросы преподавателя стоит быть уверенным, четким и логичным, опираясь на изученный материал. Также полезно поддерживать диалог

с однокурсниками, чтобы обогатить обсуждение и расширить свои знания.

Домашнее задание – набор задач по темам недели.

При работе над домашними заданиями важно внимательно ознакомиться с требованиями и сроками выполнения. Рекомендуется разбивать задания на этапы, чтобы избежать перегрузки и лучше усвоить материал, использовать различные источники информации, включая учебники и онлайн-ресурсы, для более глубокого понимания темы.

Контрольная работа – письменная работа с набором задач, которые нужно решить за ограниченное время.

Цель контрольной работы – получить специальные знания по одной или нескольким темам дисциплины и продемонстрировать навыки их практического применения.

Самостоятельная работа – работа студентов, направленная на углубленное изучение отдельных тем и вопросов учебной дисциплины (модуля).

В процессе самостоятельной работы студенты взаимодействуют с рекомендованными материалами при минимальном участии преподавателя. Задачи студента включают работу с конспектами лекций (обработка текста), повторное изучение учебных материалов планов и тезисов ответов, изучение дополнительных тем, выполнение учебно-исследовательских заданий и другое.

Система оценивания результатов обучения по дисциплине (модулю)

Критерии получения уровня и оценивания сформированности компетенций по дисциплине (модулю) «SQL и базы данных для разработчиков»

Оценивание уровня учебных достижений обучающихся по дисциплине (модулю) осуществляется в виде текущего контроля успеваемости и промежуточной аттестации.

Промежуточная аттестация по дисциплине (модулю) осуществляется в форме ***зачета с оценкой***, при этом проводится оценка компетенций, сформированных по дисциплине.

Для оценивания текущего контроля успеваемости и промежуточной аттестации используется десятибалльная шкала оценивания, которая соотносится с традиционной пятибалльной шкалой следующим образом:

Десятибалльная оценка	Пятибалльная оценка	Оценка за зачет	Общая характеристика результата обучения по дисциплине (модулю)
10	Отлично	Зачтено	Студент полностью владеет знаниями, изложенными в рабочей программе, и глубоко осмысляет дисциплину (модуль). Он самостоятельно и логически последовательно отвечает на все вопросы, акцентируя внимание на наиболее важном. Умеет анализировать, сравнивать, классифицировать, обобщать, конкретизировать и систематизировать изученный материал, выделяя ключевые моменты и устанавливая причинно-следственные связи. Четко формулирует ответы, уверенно интерпретирует результаты анализов и других исследований, а также решает сложные
9	Отлично	Зачтено	
8	Отлично	Зачтено	

Десятибалльная оценка	Пятибалльная оценка	Оценка за зачет	Общая характеристика результата обучения по дисциплине (модулю)
			задачи. Студент хорошо знаком с методами исследования, необходимыми для практической деятельности, и умеет связывать теоретические аспекты дисциплины (модуля) с практическими задачами.
7	Хорошо	Зачтено	Студент обладает знаниями предмета почти в полном объеме рабочей программы и самостоятельно, логически последовательно и всесторонне отвечает на все вопросы, акцентируя внимание на наиболее значимых моментах. Он умеет анализировать, сравнивать, классифицировать, обобщать, конкретизировать и систематизировать изученный материал, выделяя его ключевые аспекты и устанавливая причинно-следственные связи. Формулирует свои ответы, уверенно интерпретирует результаты анализов и других исследований, а также решает сложные ситуационные задачи. Студент хорошо знаком с методами исследования, необходимыми для практической деятельности, и умеет связывать теоретические аспекты предмета с практическими задачами.
6	Хорошо	Зачтено	
5	Удовлетворительно	Зачтено	Студент обладает базовыми знаниями по дисциплине (модулю), но испытывает трудности при самостоятельных ответах и использует неточные формулировки. В ходе ответов он допускает ошибки, касающиеся сути вопросов. Студент способен решать только самые простые задачи и владеет лишь минимальным набором методов исследования.
4	Удовлетворительно	Зачтено	
3	Не сдан	Не зачтено	Студент не овладел обязательным минимумом знаний по предмету и не может ответить на вопросы, даже если преподаватель задает дополнительные наводящие вопросы.
2	Не сдан	Не зачтено	
1	Не сдан	Не зачтено	

Дисциплина (модуль) «SQL и базы данных для разработчиков» оценивается следующим образом:

Активность	Вес	Описание
Аудиторная работа	15%	Активное участие в семинарах: отвечать на вопросы преподавателя и задавать свои

Электронный документ

Активность	Вес	Описание
Домашние задания	35%	Набор задач по темам недели
Контрольная работа	20%	Письменная работа с набором задач, которые нужно решить за ограниченное время
Зачет с оценкой	30%	Письменная или устная работа над заданием, направленным на проверку полученных знаний и навыков по дисциплине (модулю)

Формула расчёта итоговой оценки по дисциплине (модулю) «SQL и базы данных для разработчиков»: «0,15 × аудиторная работа + 0,35 × среднее за домашние задания + 0,2 × контрольная работа + 0,3 × зачет с оценкой».

Текущий контроль успеваемости обучающихся по дисциплине (модулю)

Примерные вопросы для подготовки к семинарам

Подготовка к семинару 1.

1. Что такое реляционная база данных и чем SQL отличается от конкретной СУБД (PostgreSQL/MySQL)? Какие основные объекты есть в БД (таблица, строка, столбец, ключ, ограничение)?

2. Каков логический порядок выполнения **SELECT** (FROM/JOIN → WHERE → GROUP BY → HAVING → SELECT → ORDER BY → LIMIT) и почему это важно при написании запросов к одной таблице?

3. В каких случаях использовать **DISTINCT**, **IN**, **BETWEEN**, **LIKE**, **IS NULL**? Какие типичные ошибки возникают из-за **NULL** в условиях фильтрации?

4. Чем отличаются агрегирование и группировка: **COUNT(*)** vs **COUNT(column)**, когда нужен **HAVING**, и как корректно агрегировать, чтобы не “потерять” строки?

5. Чем **INNER JOIN** отличается от **LEFT/RIGHT/FULL JOIN** и как соединения связаны с теоретико-множественными операциями **UNION/INTERSECT/EXCEPT**? Приведите пример задачи, где **EXCEPT** читаемее, чем **JOIN**.

Подготовка к семинару 2.

1. Когда лучше использовать подзапрос, а когда — CTE (**WITH**)? Чем отличается обычный CTE от рекурсивного, и какие плюсы/минусы (читаемость, оптимизация, повторное использование)?

2. Что такое оконные функции и чем они отличаются от агрегирования через **GROUP BY**? Какие задачи проще решать окнами (top-N по группам, скользящие метрики, дедупликация)?

3. Какие основные DDL-операции нужны разработчику: **CREATE/ALTER/DROP**, ограничения (**PK/FK/UNIQUE/CHECK**), представления (**VIEW**/**MATERIALIZED VIEW**)? В каких случаях использовать представление вместо “сложного запроса”?

4. Как читать план запроса **EXPLAIN (ANALYZE)** и какие ключевые узлы/метрики важны (Seq Scan/Index Scan, Hash Join/Nested Loop, cost, rows, actual time, buffers)? Как понять, что запрос “плохой”?

5. Какие бывают индексы (B-tree, Hash, GIN, GiST, BRIN) и под какие запросы они подходят? Как индексы влияют на скорость чтения и записи? Зачем нужны функции и процедуры в SQL, и где граница между логикой в БД и в приложении?

Подготовка к семинару 3.

1. Что такое транзакция и ACID в контексте PostgreSQL? Чем отличаются уровни изоляции **READ COMMITTED**, **REPEATABLE READ**, **SERIALIZABLE**, какие аномалии они предотвращают и какая цена за это платится?

2. Как работает MVCC в PostgreSQL: что такое “снимок данных”, версии строк, видимость изменений? Почему PostgreSQL нуждается в VACUUM и что делает

autovacuum?

3. Какие виды блокировок существуют (строчные/табличные, **ROW EXCLUSIVE**, **ACCESS SHARE** и т. п.) и как проявляются блокировки в реальных ситуациях (долгие транзакции, очереди запросов, дедлоки)? Как диагностировать блокировки через **pg_stat_activity**/**pg_locks**?

4. Что такое buffer cache и WAL (журнал предзаписи)? Как они влияют на производительность и надежность? Что означает “fsync/синхронная запись”, и какие компромиссы возможны?

5. Какие базовые подходы к масштабированию PostgreSQL существуют (реплики для чтения, partitioning, шардирование на уровне приложения)? Чем отличаются резервное копирование логическое (**pg_dump**) и физическое (**pg_basebackup**), и как связана репликация с восстановлением после сбоев?

Примерные домашние задания

Домашнее задание 1.

Задача 1.

1. Подключитесь к БД, выведите список таблиц/представлений и их столбцов (через **information_schema** или **psql \d**).
2. Напишите 10 запросов **SELECT** к **одной таблице**: фильтры **WHERE**, сортировки **ORDER BY**, **LIMIT/OFFSET**, **DISTINCT**, **IN**, **BETWEEN**, **LIKE**.
3. В **README** поясните, чем отличаются **WHERE** и **HAVING** (на уровне смысла).

Задача 2.

1. Составьте 8 запросов к одной таблице, где используются функции:
   - даты/времени (например, **date_trunc**, **extract**, **now()**),
   - текста (например, **lower/upper**, **substring**, **concat**),
   - чисел (например, **round**, **ceil/floor**).
2. Добавьте 2 примера приведения типов (**CAST**, **::type**) и объясните, зачем это нужно.

Задача 3.

1. Напишите 6 агрегирующих запросов с **GROUP BY** и агрегатами (**count/sum/avg/min/max**).
2. Используйте **HAVING** минимум в 3 запросах (фильтрация групп).
3. Сделайте один запрос для поиска “аномалий” (например, пользователи без заказов, товары без продаж — допускается через подзапросы/**JOIN** в следующем ДЗ, но здесь попробуйте в рамках одной таблицы: **COUNT**, **NULL**-поля, распределения).

Задача 4.

1. Подберите 4 бизнес-сценария (например, “заказы с данными клиента”, “состав заказа”, “платежи по заказам”, “доставка по заказам”) и реализуйте запросы через:
   - **INNER JOIN** (минимум 2),
   - **LEFT JOIN** (минимум 2).
2. В одном из запросов явно покажите проблему “размножения строк” при неверном JOIN и исправьте её.

Задача 5.

1. Реализуйте 3 запроса с **UNION/UNION ALL** и объясните разницу.
2. Реализуйте по одному запросу с **INTERSECT** и **EXCEPT** (например, “клиенты, которые покупали в 2024, но не покупали в 2025”).
3. Сравните один кейс “через JOIN” и “через EXCEPT/INTERSECT”: что читаемее/надежнее.

Домашнее задание 2.

Задача 1.

1. Сделайте 5 запросов с подзапросами: **IN**, **EXISTS**, скалярный подзапрос в **SELECT**, коррелированный подзапрос.

2. Перепишите минимум 2 запроса в виде **WITH** (CTE), чтобы улучшить читаемость.

3. Один кейс — “Top-N внутри группы” (если без оконных сложно — сделайте заготовку, доведете в ДЗ 7).

Задача 2.

1. Напишите 6 запросов с оконными функциями:

- **ROW_NUMBER** или **RANK/DENSE_RANK**,
- **LAG/LEAD**,
- оконная сумма/среднее **SUM(...) OVER (...)**.

2. Реализуйте:

- “топ-3 товара по выручке в каждой категории”,
- “накопительный итог выручки по дням”,
- “разница с предыдущей покупкой пользователя”.

Задача 3.

1. Создайте 2 таблицы (например, **app_users**, **app_events**) с ограничениями (**PK**, **FK**, **UNIQUE**, **CHECK**).

2. Выполните 5 изменений схемы: **ALTER TABLE ADD COLUMN**, изменение типа, добавление/удаление ограничения, **DROP COLUMN** (на тестовых таблицах).

3. Создайте:

- обычное **VIEW** (например, “витрина заказов”),
- при желании — **MATERIALIZED VIEW** и обновление (**REFRESH**), если хотите сравнить скорость.

Задача 4.

1. Выберите 2 “тяжелых” запроса (например, с JOIN + GROUP BY).

2. Для каждого снимите **EXPLAIN (ANALYZE, BUFFERS)** “до” и “после” оптимизации.

3. Оптимизация должна включать минимум 2 приема из списка:

- переписывание запроса (CTE/подзапрос/предикаты),
- добавление индекса,
- устранение лишних **DISTINCT**,
- изменение порядка JOIN/фильтрации.

4. В **README** опишите, что изменилось в плане (какие узлы/стоимость/время).

Задача 5.

1. Создайте и сравните индексы:

- B-tree индекс под типовой фильтр/сортировку,
- составной индекс (2 колонки),
- частичный индекс (WHERE …),
- (если есть текстовые поля) GIN индекс для поиска (**to_tsvector** или **pg_trgm**, если доступно).

2. Проверьте влияние индексирования через **EXPLAIN (ANALYZE, BUFFERS)**.

3. Напишите:

- SQL-функцию (например, **get_customer_ltv(customer_id)**),
- процедуру или функцию “пакетного обновления” (например, пересчет агрегатов/витрины за период).

Домашнее задание 3.

Задача 1.

В двух сессиях psql воспроизведите и зафиксируйте результаты:

1. При **READ COMMITTED**: пример non-repeatable read или “потерянного обновления” (если получится) / видимость изменений между SELECT.

2. При **REPEATABLE READ**: покажите, что повторный SELECT видит “снимок”.

3. При **SERIALIZABLE**: добейтесь serialization failure на конкурентном сценарии (или объясните, почему не получилось на вашем сценарии).

Сдайте **sql** со сценариями + объяснение, что наблюдали.

Задача 2.

1. На тестовой таблице сделайте массовые **INSERT/UPDATE/DELETE**.
2. Посмотрите статистику “мертвых строк” и активности по таблице (например, **pg_stat_user_tables**, **n_dead_tup**).
3. Выполните **VACUUM (ANALYZE)** и повторите измерения.
4. В **README**:

- что такое MVCC “на пальцах”,
- почему без VACUUM может деградировать производительность.

Задача 3.

1. В двух/трех сессиях воспроизведите блокировку (например, долгий **UPDATE** + другой **UPDATE**/**SELECT FOR UPDATE**).
2. Посмотрите, кто кого блокирует, через **pg_locks** + **pg_stat_activity** (выведите pid, query, locktype, relation).
3. Воспроизведите дедлок (две транзакции обновляют две строки в разном порядке), зафиксируйте ошибку.
4. Предложите 2 способа профилактики дедлоков (порядок захвата, меньшие транзакции, ретраи и т. п.).

Задача 4.

1. Подготовьте таблицу и выполните запросы, которые делают:

- последовательное чтение (Seq Scan),
- выборку по индексу (Index Scan).

2. Снимите **EXPLAIN (ANALYZE, BUFFERS)** и сравните показатели буферов.
3. Сделайте серию мелких **UPDATE/INSERT** в транзакции и сравните поведение при **synchronous_commit = on/off** (на тестовой среде) — время выполнения и смысл trade-off.
4. Коротко объясните назначение WAL и почему он важен для надежности.

Задача 5.

Выберите вариант A или B.

Вариант A (рекомендуется, если есть Docker):

1. Поднимите PostgreSQL primary + replica (streaming replication) в Docker Compose.
2. Проверьте, что на реплике доступно чтение и данные появляются после записи на primary.
3. Опишите, что будет при падении primary (без автоматического failover/с ним — теоретически).

Вариант B (если репликацию поднять сложно):

1. Сделайте бэкап **pg_dump** и восстановление **pg_restore** в новую БД.
2. Сравните логический бэкап (pg_dump) и физический (pg_basebackup) — когда какой применяют.
3. Опишите 3 подхода к масштабированию чтения/записи в Postgres (реплики для чтения, шардирование на уровне приложения, partitioning и т. п.).

Примерные задания для контрольной работы

Схема (дана):

- **customers(id PK, email, full_name, city, created_at timestamptz)**
- **products(id PK, name, category, price numeric(10,2), is_active boolean)**
- **orders(id PK, customer_id FK->customers.id, status text, created_at timestamptz)**
- **order_items(order_id FK->orders.id, product_id FK->products.id, qty int, price numeric(10,2), PRIMARY KEY(order_id, product_id))**

(price — цена на момент покупки)

- **payments(id PK, order_id FK->orders.id, amount numeric(10,2), paid_at timestamptz, method text, status text)**

Задача 1.

Напишите запросы к **одной** таблице **customers**:

a) Выведите **id, email, created_at** клиентов, зарегистрированных за последние 30 дней, отсортируйте по **created_at** по убыванию, покажите первые 20 строк.

b) Найдите клиентов, у которых **email** оканчивается на **@gmail.com** и город не **NULL**.

c) Выведите уникальные города (**DISTINCT city**) и количество клиентов в каждом городе (можно без JOIN).

Задача 2.

Таблица **products(price numeric(10,2), is_active boolean)**.

a) Выведите **name**, **price**, а также **price** в целых рублях (округление до целого) и **price** как текст в формате **'RUB: <значение>'**.

b) Выведите только активные товары (**is_active = true**), у которых цена строго больше **1000**.

c) Приведите пример запроса, где явное приведение типа (**CAST** или **::**) нужно, чтобы сравнение/конкатенация работали корректно (на данных из этой схемы).

Задача 3.

Считайте выручку по позициям заказа как **qty * order_items.price**.

a) Посчитайте выручку по категориям товаров (**products.category**) за 2025 год.

b) Оставьте только категории, где выручка > 1 000 000 (используйте **HAVING**).

c) Для каждой категории дополнительно выведите количество уникальных заказов.

Задача 4.

Сформируйте отчет по заказам:

a) Для каждого заказа выведите: **order_id**, **created_at**, **customer email**, **status**, количество позиций (сумма **qty**), сумма заказа.

b) Включите в отчет заказы даже без оплат (если такие есть).

c) Добавьте в отчет признак **is_paid**: true, если есть хотя бы один платеж со **status = 'succeeded'**.

Задача 5.

Пусть “активный клиент” — клиент, сделавший заказ.

a) Список **customer_id**, которые делали заказы **в январе 2025** или **в феврале 2025** (используйте **UNION** или **UNION ALL** с объяснением выбора).

b) Список **customer_id**, которые делали заказы **и в январе, и в феврале 2025** (используйте **INTERSECT**).

c) Список **customer_id**, которые делали заказы в январе 2025, но **не** делали в феврале 2025 (используйте **EXCEPT**).

Задача 6.

a) Найдите клиентов, у которых сумма всех успешных оплат (**payments.status='succeeded'**) выше среднего по всем клиентам (подзапрос со средним).

b) Перепишите пункт (a) через **WITH** (CTE), разбив на 2–3 логических шага (например: **customer_revenue**, **avg_revenue**, итоговая выборка).

Задача 7.

a) Для каждого клиента выведите список его заказов: **customer_id, order_id, created_at**, номер заказа по времени (**row_number()** по **created_at**).

b) Для каждого клиента посчитайте “накопительную сумму оплат” по времени оплат (**SUM(amount) OVER (PARTITION BY customer_id ORDER BY paid_at)**), учитывая только **succeeded**.

c) Выведите **топ-3 клиентов** по сумме успешных оплат в каждом городе (**RANK()** или **DENSE_RANK()** по городу).

Задача 8.

a) Создайте таблицу **deliveries**:

- **id bigserial PK**
- **order_id** FK на **orders(id)** (уникальный, один заказ — одна доставка)
- **courier_name text NOT NULL**
- **status text NOT NULL** (ограничьте **CHECK** на значения:

('created','in_transit','delivered','canceled'))

- **created_at timestamptz NOT NULL DEFAULT now()**

b) Создайте **VIEW v_order_full** со сводной информацией: **order_id**, **customer_email**, **order_created_at**, **order_sum**, **paid_sum**, **is_fully_paid** (paid_sum >= order_sum).

Задача 9.

Дан запрос, который выполняется медленно:

```
SELECT o.id, o.created_at
FROM orders o
WHERE o.customer_id = :cid
  AND o.created_at >= :from_dt
ORDER BY o.created_at DESC
LIMIT 50;
```

a) Предложите индекс(ы), которые улучшат запрос, и обоснуйте порядок полей в индексе.

b) Какой тип скана/узел плана вы ожидаете увидеть в **EXPLAIN** после добавления индекса (например, Index Scan/Bitmap Index Scan)? Почему?

c) Назовите 2 причины, почему индекс мог не использоваться (статистика, селективность, тип условия, настройки и т.п.).

Задача 10.

Ответьте (можно текстом + при необходимости SQL-команды):

a) Две сессии. В сессии A: **BEGIN; SELECT ...** читает строку заказа. В сессии B: **BEGIN; UPDATE orders SET status='canceled' WHERE id=...; COMMIT;** — Что увидит повторный **SELECT** в сессии A при **READ COMMITTED** и при **REPEATABLE READ**? Объясните через MVCC/снимок данных.

b) Что такое autovacuum и какие проблемы возникают при длительных транзакциях (связь с “мертвыми” версиями строк и ростом таблиц)?

c) Приведите пример сценария дедлока на двух таблицах/строках и 2 способа его предотвращения.

d) Для чего нужен WAL и чем логический бэкап (**pg_dump**) отличается от физического (например, **pg_basebackup**)? Когда что применяют?

e) Опишите базовую схему репликации “primary → read replica” и типичный вариант масштабирования чтения в PostgreSQL.

Задания для промежуточной аттестации по дисциплине (модулю)

№ п/п	Задание	Ответ	Компетенция
1.	Укажите номер и название SQL-объекта, который создают для серверной логики (вычисления на стороне БД) и вызывают из запросов. Варианты: 1 FUNCTION; 2 DATABASE; 3 SCHEMA.	1 FUNCTION / 1 Функция	ОПК-3
2.	Укажите номер и название части SELECT-запроса, которая фильтрует строки до группировки. Варианты: 1 WHERE; 2 HAVING; 3 ORDER BY.	1 WHERE	ПК-3
3.	Укажите номер и название утилиты PostgreSQL для логического резервного копирования базы данных. Варианты: 1 pg_dump; 2 psql; 3 pg_ctl.	1 pg_dump	ПК-6
4.	Укажите номер и название типа индекса PostgreSQL, который обычно выбирают для полнотекстового поиска/TSVECTOR. Варианты: 1 B-tree; 2 GIN; 3 Hash.	2 GIN	ОПК-3
5.	Укажите номер и название части SELECT-запроса, которая фильтрует группы после GROUP BY. Варианты: 1 WHERE; 2 HAVING; 3 FROM.	2 HAVING	ПК-3

6.	Укажите номер и название компонента PostgreSQL, который используется для восстановления после сбоя и репликации (журнал предзаписи). Варианты: 1 WAL; 2 Buffer cache; 3 CHECK.	1 WAL	ПК-6
7.	Укажите номер и название команды PostgreSQL, которую используют для измерения реального времени выполнения SQL-запроса по плану. Варианты: 1 EXPLAIN; 2 EXPLAIN ANALYZE; 3 VACUUM.	2 EXPLAIN ANALYZE / 2 EXPLAIN (ANALYZE)	ОПК-3
8.	Укажите номер и название типа JOIN, который оставляет все строки из левой таблицы, даже если справа нет совпадений. Варианты: 1 INNER JOIN; 2 LEFT JOIN; 3 CROSS JOIN.	2 LEFT JOIN / 2 LEFT OUTER JOIN	ПК-3
9.	Укажите номер и название фонового процесса PostgreSQL, который выполняет автоочистку “мёртвых” версий строк. Варианты: 1 autovacuum; 2 autocommit; 3 autoscale.	1 autovacuum	ПК-6
10.	Укажите номер и название объекта SQL, который создают для переиспользования запроса как логического представления данных. Варианты: 1 VIEW; 2 INDEX; 3 TRIGGER.	1 VIEW / 1 Представление	ОПК-3
11.	Укажите номер и название теоретико-множественной операции SQL для задачи: «вернуть строки, которые есть в A, но отсутствуют в B». Варианты: 1 UNION; 2 INTERSECT; 3 EXCEPT.	3 EXCEPT	ПК-3
12.	Укажите номер и название механизма PostgreSQL, который обеспечивает многоверсионность и чтение «снимка данных». Варианты: 1 MVCC; 2 WAL; 3 CDN.	1 MVCC	ПК-6
13.	Укажите номер и название SQL-оператора, который изменяет структуру существующей таблицы (например, добавить столбец). Варианты: 1 UPDATE; 2 ALTER TABLE; 3 INSERT.	2 ALTER TABLE	ОПК-3
14.	Укажите номер и название теоретико-множественной операции SQL для задачи: «вернуть строки, которые есть и в A, и в B». Варианты: 1 UNION; 2 INTERSECT; 3 EXCEPT.	2 INTERSECT	ПК-3
15.	Укажите номер и название SQL-команды, которую применяют в транзакции PostgreSQL, чтобы заблокировать выбранные строки на обновление. Варианты: 1 SELECT FOR UPDATE; 2 SELECT DISTINCT; 3 SELECT UNION.	1 SELECT FOR UPDATE	ПК-6
16.	Укажите номер и название SQL-оператора, который создаёт таблицу. Варианты: 1 CREATE TABLE; 2 ALTER TABLE; 3 DROP TABLE.	1 CREATE TABLE	ОПК-3
17.	Укажите номер и название агрегатной функции SQL для задачи: «посчитать количество строк с учётом NULL». Варианты: 1 COUNT(*); 2 COUNT(column); 3 SUM(column).	1 COUNT(*)	ПК-3
18.	Укажите номер и название уровня изоляции по умолчанию в PostgreSQL. Варианты: 1 READ UNCOMMITTED; 2 READ COMMITTED; 3 REPEATABLE READ; 4 SERIALIZABLE.	2 READ COMMITTED	ПК-6