*Санкт-Петербургский Национальный Исследовательский Университет Информационных Технологий, Механики и Оптики*

*ПИиКТ*

*Лабораторная работа 3*

*по дисциплине*

*«Информационные системы и базы данных»*

*Выполнил: Студент группы P33113*

*Мансуров Б.Б.*

*Преподаватель: Николаев В.В.*

*Санкт-Петербург*

*2020г*

# *Задание*

*Составить запросы на языке SQL (пункты 1-2).*

*Для каждого запроса предложить индексы, добавление которых уменьшит время выполнения запроса (указать таблицы/атрибуты, для которых нужно добавить индексы, написать тип индекса; объяснить, почему добавление индекса будет полезным для данного запроса).*

*Для запросов 1-2 необходимо составить возможные планы выполнения запросов. Планы составляются на основании предположения, что в таблицах отсутствуют индексы. Из составленных планов необходимо выбрать оптимальный и объяснить свой выбор.  
Изменятся ли планы при добавлении индекса и как?*

*Для запросов 1-2 необходимо добавить в отчет вывод команды EXPLAIN ANALYZE [запрос]*

*Подробные ответы на все вышеперечисленные вопросы должны присутствовать в отчете (планы выполнения запросов должны быть нарисованы, ответы на вопросы - представлены в текстовом виде).*

***ЗАПРОС 1***

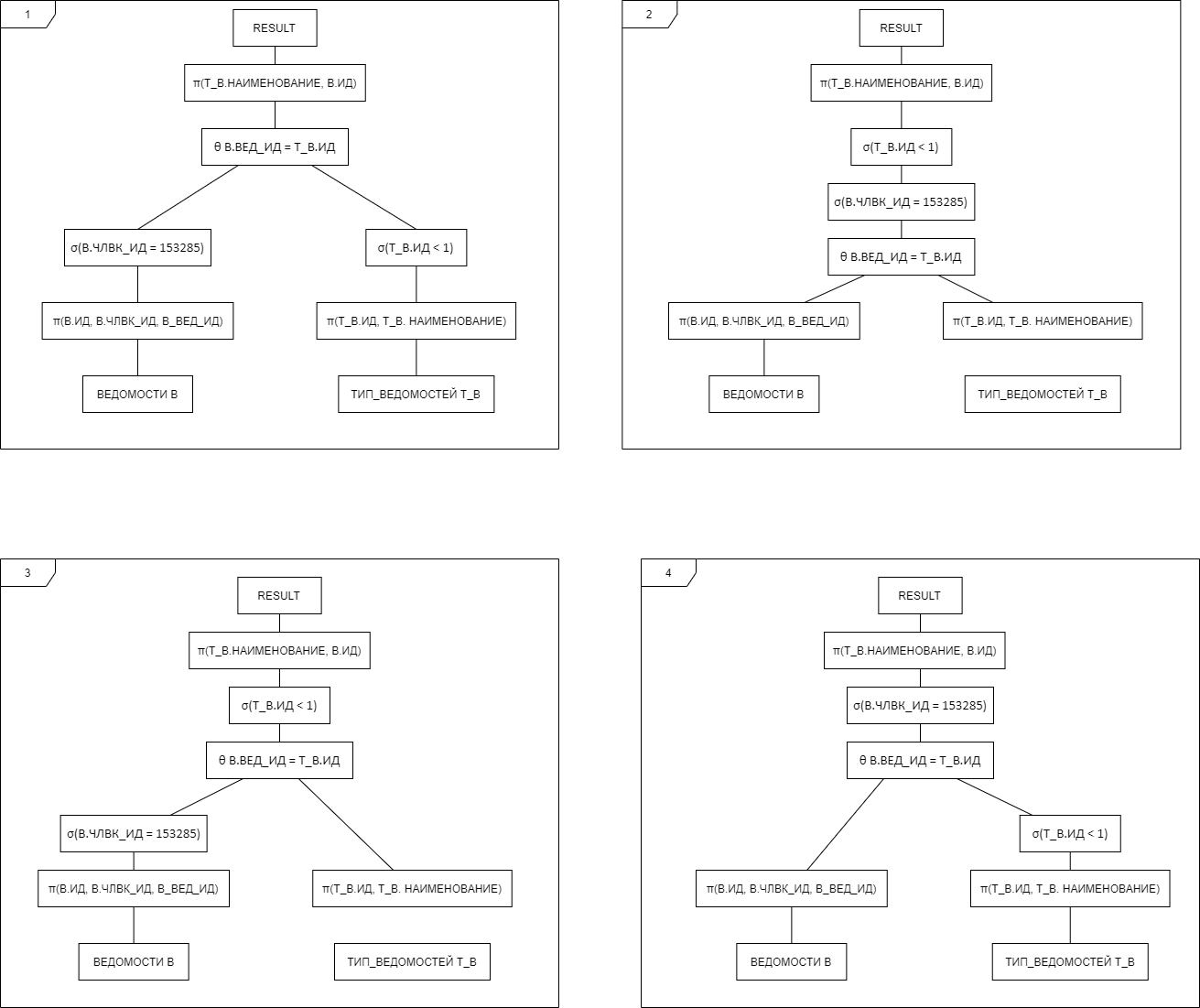
*-- 1. Сделать запрос для получения атрибутов из указанных таблиц, применив фильтры по указанным условиям:  
-- Н\_ТИПЫ\_ВЕДОМОСТЕЙ, Н\_ВЕДОМОСТИ.  
-- Вывести атрибуты: Н\_ТИПЫ\_ВЕДОМОСТЕЙ.НАИМЕНОВАНИЕ, Н\_ВЕДОМОСТИ.ИД.  
-- Фильтры (AND):  
-- a) Н\_ТИПЫ\_ВЕДОМОСТЕЙ.ИД <1.  
-- b) Н\_ВЕДОМОСТИ.ЧЛВК\_ИД > 117219.  
-- c) Н\_ВЕДОМОСТИ.ЧЛВК\_ИД = 153285.  
-- Вид соединения: RIGHT JOIN.*

*SELECT ТИПЫ\_ВЕДОМОСТЕЙ.НАИМЕНОВАНИЕ, ВЕДОМОСТИ.ИД  
FROM Н\_ТИПЫ\_ВЕДОМОСТЕЙ AS ТИПЫ\_ВЕДОМОСТЕЙ  
 RIGHT JOIN Н\_ВЕДОМОСТИ AS ВЕДОМОСТИ ON ТИПЫ\_ВЕДОМОСТЕЙ.ИД = ВЕДОМОСТИ.ВЕД\_ИД  
WHERE ТИПЫ\_ВЕДОМОСТЕЙ.ИД < 1  
 AND ВЕДОМОСТИ.ЧЛВК\_ИД > 117219  
 AND ВЕДОМОСТИ.ЧЛВК\_ИД = 153285;*

*Первое что мы можем сделать для оптимизации убрать ВЕДОМОСТИ.ЧЛВК\_ИД > 117219. Потому что после этого у нас идет выборка по “=”, и в нашем случае лучше не делать лишний операции выборки. После изменений мы получим. Я проверил с помощью EXPLAIN ANALYZE.*

*SELECT ТИПЫ\_ВЕДОМОСТЕЙ.НАИМЕНОВАНИЕ, ВЕДОМОСТИ.ИД  
FROM Н\_ТИПЫ\_ВЕДОМОСТЕЙ AS ТИПЫ\_ВЕДОМОСТЕЙ  
 RIGHT JOIN Н\_ВЕДОМОСТИ AS ВЕДОМОСТИ ON ТИПЫ\_ВЕДОМОСТЕЙ.ИД = ВЕДОМОСТИ.ВЕД\_ИД  
WHERE ТИПЫ\_ВЕДОМОСТЕЙ.ИД < 1  
 AND ВЕДОМОСТИ.ЧЛВК\_ИД = 153285;*

***ПЛАНЫ ВЫПОЛНЕНИЯ ЗАПРОСА***

**

*Чтобы оптимизировать запрос, необходимо делать выборку как можно раньше. Оптимальным является план номер 1. Если бы мы оставили Н\_ВЕДОМОСТИ.ЧЛВК\_ИД > 117219. Имеет смысл сделать сначала выборку по Н\_ВЕДОМОСТЬ.ЧЛВК\_ИД = 153285, а потом >, так как мы сильно сузим множество и не надо будет 2 раза по всем ЧЛВЕ\_ИД.*

***ИНДЕКСЫ***

*Т\_В.ИД добавить индекса смысла нет, так как индексы неэффективны если в таблице мало строк (В таблице Т\_В 3 строк), и они будут занимать место.*

*На В.ЧЛВК\_ИД имеет смысл добавить индекс. Если мы оставим Н\_ВЕДОМОСТИ.ЧЛВК\_ИД > 117219 то нужно добавить B\_TREE (дерево спускается по веткам на основе сравнений, и работает за log(глубина), а обычный поиск за линейное время), так как будет использоваться и =, и >.*

*Если мы не оставим Н\_ВЕДОМОСТИ.ЧЛВК\_ИД > 11721 то лучше добавить HASH*

*Добавление*

*CREATE INDEX ИН\_ВЕД\_ЧЛВК\_ИД ON "Н\_ВЕДОМОСТИ" USING btree ("ЧЛВК\_ИД");*

***или*** *CREATE INDEX ИН\_ВЕД\_ЧЛВК\_ИД ON "Н\_ВЕДОМОСТИ" USING hash ("ЧЛВК\_ИД");*

*Этих индексов может ускорить запросы, потому что по данным атрибутам идет выборка с использованием операторов = и >.*

***EXPLAIN ANALYZE***

*QUERY PLAN  
---------------------------------------------------------------------------------------------------------------------------------------  
 Nested Loop (cost=0.42..197.94 rows=1 width=422) (actual time=0.024..0.024 rows=0 loops=1)  
 Join Filter: ("ТИПЫ\_ВЕДОМОСТЕЙ"."ИД" = "ВЕДОМОСТИ"."ВЕД\_ИД")  
 -> Seq Scan on "Н\_ТИПЫ\_ВЕДОМОСТЕЙ" "ТИПЫ\_ВЕДОМОСТЕЙ" (cost=0.00..1.04 rows=1 width=422) (actual time=0.023..0.023 rows=0 loops=1)  
 Filter: ("ИД" < 1)  
 Rows Removed by Filter: 3  
 -> Index Scan using "ВЕД\_ЧЛВК\_FK\_IFK" on "Н\_ВЕДОМОСТИ" "ВЕДОМОСТИ" (cost=0.42..196.10 rows=64 width=8) (never executed)  
 Index Cond: ("ЧЛВК\_ИД" = 153285)  
 Planning time: 0.204 ms  
 Execution time: 0.060 ms  
(9 rows)* ***Или***

*QUERY PLAN  
---------------------------------------------------------------------------------------------------------------------------------------  
 Nested Loop (cost=0.42..161.96 rows=1 width=422) (actual time=0.032..0.032 rows=0 loops=1)  
 Join Filter: ("ТИПЫ\_ВЕДОМОСТЕЙ"."ИД" = "ВЕДОМОСТИ"."ВЕД\_ИД")  
 -> Seq Scan on "Н\_ТИПЫ\_ВЕДОМОСТЕЙ" "ТИПЫ\_ВЕДОМОСТЕЙ" (cost=0.00..1.04 rows=1 width=422) (actual time=0.030..0.030 rows=0 loops=1)  
 Filter: ("ИД" < 1)  
 Rows Removed by Filter: 3  
 -> Index Scan using "ВЕД\_ЧЛВК\_FK\_IFK" on "Н\_ВЕДОМОСТИ" "ВЕДОМОСТИ" (cost=0.42..160.28 rows=52 width=8) (never executed)  
 Index Cond: (("ЧЛВК\_ИД" > 117219) AND ("ЧЛВК\_ИД" = 153285))  
 Planning time: 0.381 ms  
 Execution time: 0.095 ms  
(9 rows)*

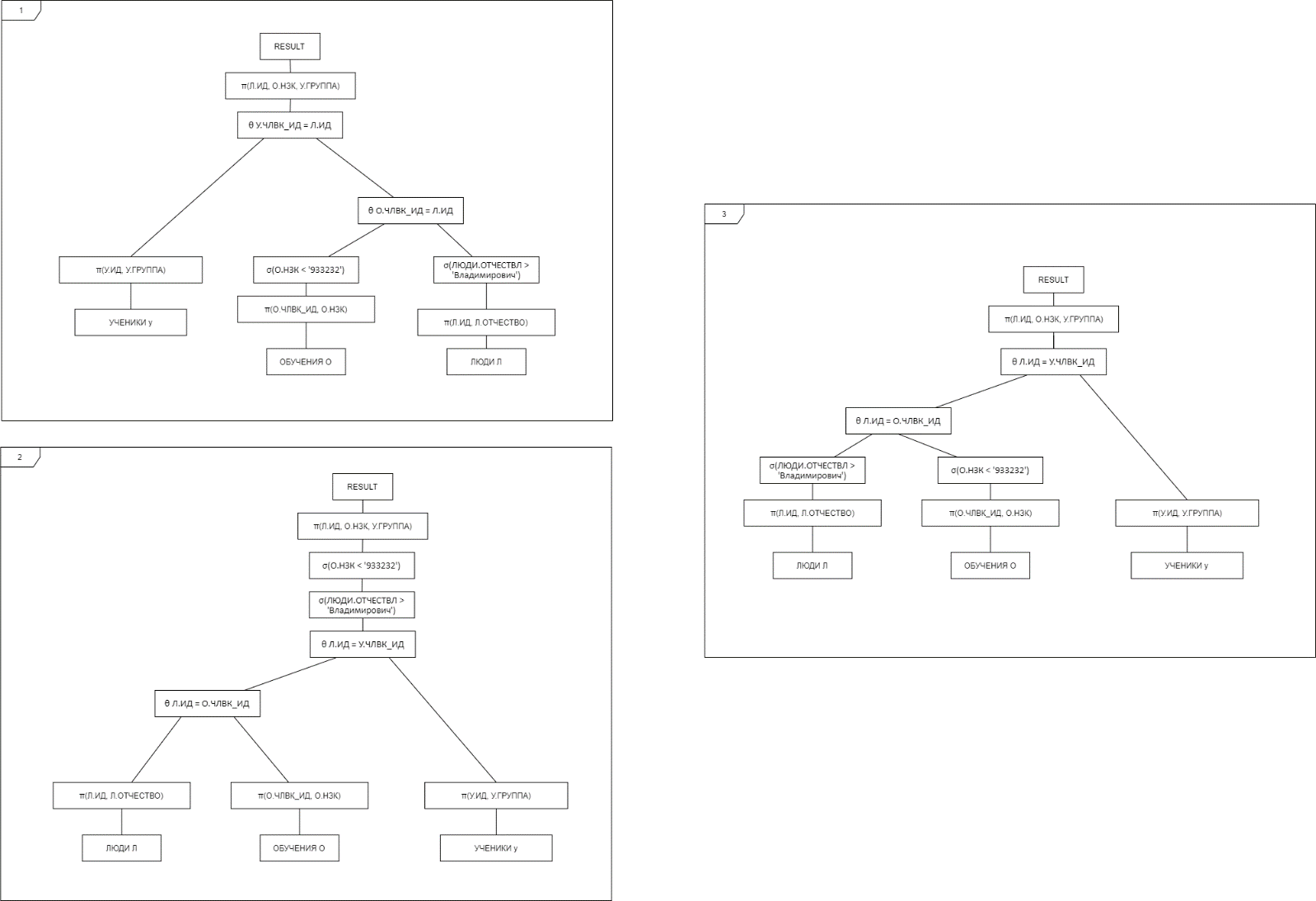
***ЗАПРОС 2***

*-- 2. Сделать запрос для получения атрибутов из указанных таблиц, применив фильтры по указанным условиям:  
-- Таблицы: Н\_ЛЮДИ, Н\_ОБУЧЕНИЯ, Н\_УЧЕНИКИ.  
-- Вывести атрибуты: Н\_ЛЮДИ.ИД, Н\_ОБУЧЕНИЯ.НЗК, Н\_УЧЕНИКИ.ГРУППА.  
-- Фильтры: (AND)  
-- a) Н\_ЛЮДИ.ОТЧЕСТВО > Владимирович.  
-- b) Н\_ОБУЧЕНИЯ.НЗК < 933232.  
-- Вид соединения: LEFT JOIN.*

*SELECT ЛЮДИ.ИД, ОБУЧЕНИЯ.НЗК, УЧЕНИКИ.ГРУППА  
FROM Н\_ЛЮДИ AS ЛЮДИ  
 LEFT JOIN Н\_ОБУЧЕНИЯ AS ОБУЧЕНИЯ ON ЛЮДИ.ИД = ОБУЧЕНИЯ.ЧЛВК\_ИД  
 LEFT JOIN Н\_УЧЕНИКИ AS УЧЕНИКИ ON ЛЮДИ.ИД = УЧЕНИКИ.ЧЛВК\_ИД  
WHERE ЛЮДИ.ОТЧЕСТВО > 'Владимирович'  
 AND ОБУЧЕНИЯ.НЗК < '933232';*

*Можно улучшить этот запрос. Потому что*

*EXPLAIN ANALYSE  
SELECT ЛЮДИ.ИД, ОБУЧЕНИЯ.НЗК, УЧЕНИКИ.ГРУППА  
FROM Н\_УЧЕНИКИ AS УЧЕНИКИ  
 RIGHT JOIN Н\_ЛЮДИ AS ЛЮДИ ON УЧЕНИКИ.ЧЛВК\_ИД = ЛЮДИ.ИД  
 RIGHT JOIN Н\_ОБУЧЕНИЯ AS ОБУЧЕНИЯ ON ОБУЧЕНИЯ.ЧЛВК\_ИД = ЛЮДИ.ИД  
WHERE ЛЮДИ.ОТЧЕСТВО > 'Владимирович'  
 AND ОБУЧЕНИЯ.НЗК < '933232';*

***ПЛАНЫ ВЫПОЛНЕНИЯ ЗАПРОСА***

*Второй план самый неоптимальный, так как операции выборки происходят после объединений.  
Третий план не самый оптимальный потому что если у нас будет hash join как правило, число строк в том наборе, на основе которого строится хеш-таблица, меньше, чем во втором наборе. Это позволяет уменьшить ее размер и ускорить процесс обращения к ней. Оптимальным планом выполнения запроса является 1 так как происходит объединение только после необходимой выборки, вместо полного объединения таблиц и если будет hash join то будет построена hash-таблица по таблице в которой меньше строк.*

***ИНДЕКСЫ***

*Имеет смысл добавить на ЛЮДИ.ИД HASH – индекс (хеширование значений идет по этому ключу, таким образом время выполнения +- константное), так как идет прямое сравнение и ЛЮДИ является правой таблицей в RIGHT JOIN-е и по ней будет поиск.*

*Нужно добавить BTREE –индекс на ЛЮДИ.ОТЧЕСТВО так как при выборку используется>   
Аналогично нужно добавить BTREE – индекс на ОБУЧЕНИЯ.НЗК потому что у нас есть выборка связанное с>*

*CREATE INDEX ИН\_ЛЮДИ\_ИД ON "Н\_ЛЮДИ" USING hash ("ИД");  
CREATE INDEX ИН\_ЛЮДИ\_ОТЧЕСТВО ON "Н\_ЛЮДИ" USING btree ("ОТЧЕСТВО");  
CREATE INDEX ИН\_ОБУЧ\_НЗК ON "Н\_ОБУЧЕНИЯ" USING btree ("НЗК");*

*Добавление этих индексов может здорово ускорить выполнения запросов, потому что по данным атрибутом идет выборка и соединение таблиц.*

***EXPLAIN ANALYZE***

*QUERY PLAN  
--------------------------------------------------------------------------------------------------------------------------------------------  
 Hash Right Join (cost=365.53..1302.85 rows=7579 width=14) (actual time=8.617..26.901 rows=7226 loops=1)  
 Hash Cond: ("УЧЕНИКИ"."ЧЛВК\_ИД" = "ЛЮДИ"."ИД")  
 -> Seq Scan on "Н\_УЧЕНИКИ" "УЧЕНИКИ" (cost=0.00..774.11 rows=23311 width=8) (actual time=0.005..7.651 rows=23311 loops=1)  
 -> Hash (cost=344.73..344.73 rows=1664 width=10) (actual time=8.597..8.597 rows=1676 loops=1)  
 Buckets: 2048 Batches: 1 Memory Usage: 87kB  
 -> Hash Join (cost=195.78..344.73 rows=1664 width=10) (actual time=4.184..7.992 rows=1676 loops=1)  
 Hash Cond: ("ОБУЧЕНИЯ"."ЧЛВК\_ИД" = "ЛЮДИ"."ИД")  
 -> Seq Scan on "Н\_ОБУЧЕНИЯ" "ОБУЧЕНИЯ" (cost=0.00..119.76 rows=3347 width=10) (actual time=0.015..2.145 rows=3347 loops=1)  
 Filter: (("НЗК")::text < '933232'::text)  
 Rows Removed by Filter: 1674  
 -> Hash (cost=163.97..163.97 rows=2544 width=4) (actual time=4.126..4.126 rows=2546 loops=1)  
 Buckets: 4096 Batches: 1 Memory Usage: 122kB  
 -> Seq Scan on "Н\_ЛЮДИ" "ЛЮДИ" (cost=0.00..163.97 rows=2544 width=4) (actual time=0.015..3.015 rows=2546 loops=1)  
 Filter: (("ОТЧЕСТВО")::text > 'Владимирович'::text)  
 Rows Removed by Filter: 2572  
 Planning time: 0.505 ms  
 Execution time: 28.729 ms  
(17 rows)*

***ВЫВОД***

*Выполнив эту лабораторную работу я узнал много нового о возможностях PostgreSQL (SQL). Оказывается, если начать хорошо разбираться в вопросах оптимизации можно не хило поднят производительность если конечно все сделать грамотно. Это лабораторная работа открыла мне глаза.*