

## **Вариант 8**

### **Цель:**

Данная лабораторная работа призвана сформировать у студента понимание особенностей хранения данных приложения в РСУБД, а также настройка и поддержка хранения данных.

### **Задачи:**

- Получить теоретические знания по концептуальным картам.
- Ознакомится с понятием нормализации в БД.
- Изучить типы связей между сущностями и таблицами БД.
- Ознакомится с операторами создания БД.
- Изучение типов данных.
- Научится добавлять записи в таблицы.
- Научиться удалять и изменять записи в таблице.
- Ознакомиться с механизмами контроля согласованности БД, транзакциями и триггерами.

### **Практическое задание №1**

#### **Задание:**

К первоначальной модели предметной области необходимо добавить не менее 2-х дополнительных сущностей (таблиц), необходимых для более полного решения поставленной задачи. Необходимо создать концептуальную схему БД (в виде ER-диаграммы, содержащей таблицы и связи между ними, с уточнением типов полей, с описанием внешних и первичных ключей). При сдаче задания студент должен обосновать соответствие созданной схемы поставленной задаче.

## Предметная область для практических заданий: Управление проектом

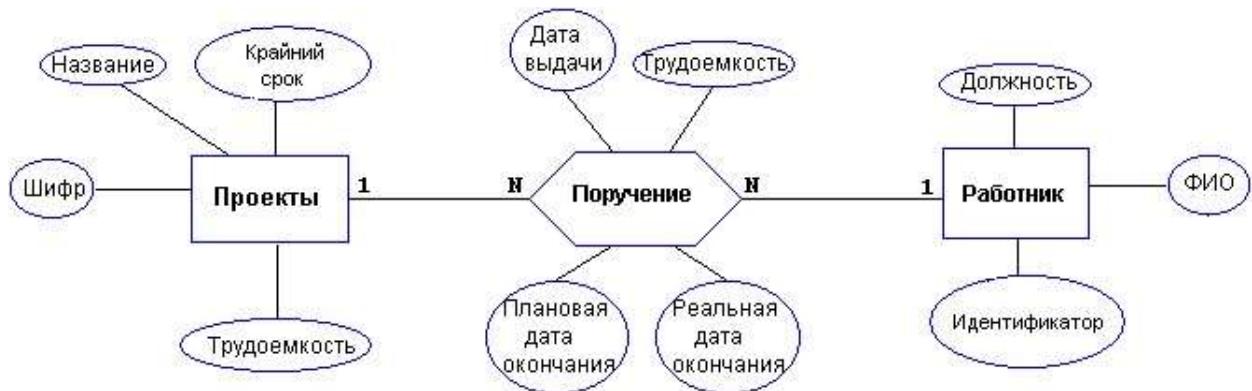


Рисунок 1.1 – предметная область

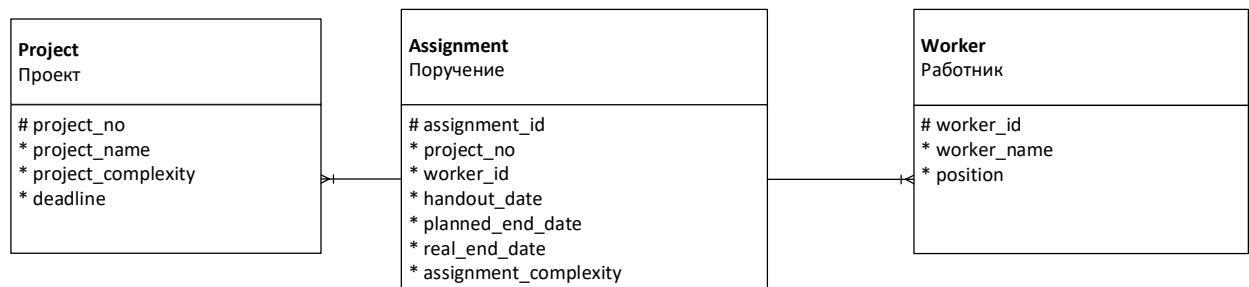


Рисунок 1.2 – ER-диаграмма

### Практическое задание №2

#### **Задание:**

Необходимо определить первичные и внешние ключи, а также ограничения полей (возможность принимать неопределенное значение, уникальные ключи, проверочные ограничения и т. д.). Таблицы следует создавать в отдельной базе данных.

```

create table assignment
(assignment_id integer not null generated always as identity,
project_no integer not null,
worker_id integer not null,
handout_date date not null,
planned_end_date date not null,
real_end_date date not null,
assignment_complexity double precision not null,
primary key (assignment_id),

```

```
foreign key(project_no)references project(project_no),  
foreign key(worker_id)references worker(worker_id));
```

Кроме того, нужно подготовить данные для заполнения созданных таблиц. Объем подготовленных данных должен составлять не менее 10 экземпляров для каждой из стержневых сущностей и 1000 экземпляров для целевой сущности. На основе этих данных необходимо создать SQL-скрипт для вставки соответствующих строк в таблицы БД.

Необходимо подготовить два запроса:

- Запрос к одной таблице, содержащий фильтрацию по нескольким полям.
- Запрос к нескольким связанным таблицам, содержащий фильтрацию по некоторым полям.

Для каждого из этих запросов необходимо провести следующие шаги:

- Получить план выполнения запроса без использования индексов.
- Получить статистику (IO и Time) выполнения запроса без использования индексов.
- Создать нужные индексы, позволяющие ускорить запрос.
- Получить статистику выполнения запроса с использованием индексов и сравнить с первоначальной статистикой.

Первый запрос к таблице assignment, содержащий фильтрацию по некоторым полям:

```
select * from assignment where worker_id=2 and project_no=1;
```

Второй запрос к таблицам assignment и worker, содержащий фильтрацию по некоторым полям:

```
select a.assignment_id,  
       a.project_no,  
       a.worker_id,  
       worker.worker_name,  
       worker.position,  
       a.assignment_complexity
```

from assignment as a

inner join worker using (worker\_id) – on a.worker\_id=worker.worker\_id

where assignment\_complexity between '10' and '20'

and worker.position = 'дизайнер'

order by assignment\_complexity;

```
lab2=# select * from assignment where worker_id=2 and project_no=1;
assignment_id | project_no | worker_id | handout_date | planned_end_date | real_end_date | assignment_complexity
+-----+-----+-----+-----+-----+-----+-----+
 165 |     1 |     2 | 2019-08-22 | 2020-03-25 | 2023-02-06 |          85
 428 |     1 |     2 | 2018-10-05 | 2022-06-16 | 2024-12-23 |          84
 445 |     1 |     2 | 2020-03-07 | 2020-11-17 | 2023-07-04 |          42
 795 |     1 |     2 | 2019-12-10 | 2021-12-19 | 2023-04-02 |          36
 830 |     1 |     2 | 2019-11-11 | 2021-07-21 | 2024-10-28 |          78
 848 |     1 |     2 | 2018-04-08 | 2021-12-16 | 2025-01-05 |          32
(6 ёсёлъ)

Время: 3,546 мс
```

Рисунок 2.1 – время выполнения первого запроса без индексов

```
lab2=# create index worker_id on assignment(worker_id);
CREATE INDEX
Время: 26,677 мс
lab2=# select * from assignment where worker_id=2 and project_no=1;
assignment_id | project_no | worker_id | handout_date | planned_end_date | real_end_date | assignment_complexity
+-----+-----+-----+-----+-----+-----+-----+
 165 |     1 |     2 | 2019-08-22 | 2020-03-25 | 2023-02-06 |          85
 428 |     1 |     2 | 2018-10-05 | 2022-06-16 | 2024-12-23 |          84
 445 |     1 |     2 | 2020-03-07 | 2020-11-17 | 2023-07-04 |          42
 795 |     1 |     2 | 2019-12-10 | 2021-12-19 | 2023-04-02 |          36
 830 |     1 |     2 | 2019-11-11 | 2021-07-21 | 2024-10-28 |          78
 848 |     1 |     2 | 2018-04-08 | 2021-12-16 | 2025-01-05 |          32
(6 ёсёлъ)

Время: 2,941 мс
```

Рисунок 2.2 – время выполнения первого запроса с индексами

```

lab2=# select assignment.assignment_id,
lab2# assignment.project_no,
lab2# assignment.worker_id,
lab2# worker.worker_name,
lab2# worker.position,
lab2# assignment.assignment_complexity from assignment
lab2# inner join worker on assignment.worker_id=worker.worker_id
lab2# where assignment_complexity between '10' and '20'
lab2# and worker.position = 'дизайнер'
lab2# order by assignment_complexity;
+-----+-----+-----+-----+-----+
| assignment_id | project_no | worker_id | worker_name | position |
+-----+-----+-----+-----+-----+
| 765 | 7 | 1 | Сидоров Олег Владимирович | дизайнер |
| 309 | 7 | 1 | Сидоров Олег Владимирович | дизайнер |
| 274 | 9 | 1 | Сидоров Олег Владимирович | дизайнер |
| 613 | 2 | 1 | Сидоров Олег Владимирович | дизайнер |
| 504 | 5 | 1 | Сидоров Олег Владимирович | дизайнер |
| 936 | 6 | 1 | Сидоров Олег Владимирович | дизайнер |
| 633 | 4 | 1 | Сидоров Олег Владимирович | дизайнер |
| 853 | 5 | 1 | Сидоров Олег Владимирович | дизайнер |
| 792 | 6 | 1 | Сидоров Олег Владимирович | дизайнер |
| 872 | 5 | 1 | Сидоров Олег Владимирович | дизайнер |
| 610 | 7 | 1 | Сидоров Олег Владимирович | дизайнер |
| 355 | 5 | 1 | Сидоров Олег Владимирович | дизайнер |
| 835 | 6 | 1 | Сидоров Олег Владимирович | дизайнер |
| 236 | 10 | 1 | Сидоров Олег Владимирович | дизайнер |
| 9 | 3 | 1 | Сидоров Олег Владимирович | дизайнер |
| 890 | 1 | 1 | Сидоров Олег Владимирович | дизайнер |
+-----+-----+-----+-----+-----+
(16 строк)

```

Время: 4,572 мс

Рисунок 2.3 – время выполнения второго запроса без индексов

```

lab2=# create index position on worker (position);
CREATE INDEX
Время: 2,687 мс
lab2# select assignment.assignment_id,
lab2# assignment.project_no,
lab2# assignment.worker_id,
lab2# worker.worker_name,
lab2# worker.position,
lab2# assignment.assignment_complexity from assignment
lab2# inner join worker on assignment.worker_id=worker.worker_id
lab2# where assignment_complexity between '10' and '20'
lab2# and worker.position = 'дизайнер'
lab2# order by assignment_complexity;
+-----+-----+-----+-----+-----+
| assignment_id | project_no | worker_id | worker_name | position |
+-----+-----+-----+-----+-----+
| 765 | 7 | 1 | Сидоров Олег Владимирович | дизайнер |
| 309 | 7 | 1 | Сидоров Олег Владимирович | дизайнер |
| 274 | 9 | 1 | Сидоров Олег Владимирович | дизайнер |
| 613 | 2 | 1 | Сидоров Олег Владимирович | дизайнер |
| 504 | 5 | 1 | Сидоров Олег Владимирович | дизайнер |
| 936 | 6 | 1 | Сидоров Олег Владимирович | дизайнер |
| 633 | 4 | 1 | Сидоров Олег Владимирович | дизайнер |
| 853 | 5 | 1 | Сидоров Олег Владимирович | дизайнер |
| 792 | 6 | 1 | Сидоров Олег Владимирович | дизайнер |
| 872 | 5 | 1 | Сидоров Олег Владимирович | дизайнер |
| 610 | 7 | 1 | Сидоров Олег Владимирович | дизайнер |
| 355 | 5 | 1 | Сидоров Олег Владимирович | дизайнер |
| 835 | 6 | 1 | Сидоров Олег Владимирович | дизайнер |
| 236 | 10 | 1 | Сидоров Олег Владимирович | дизайнер |
| 9 | 3 | 1 | Сидоров Олег Владимирович | дизайнер |
| 890 | 1 | 1 | Сидоров Олег Владимирович | дизайнер |
+-----+-----+-----+-----+-----+
(16 строк)

```

Время: 1,981 мс

Рисунок 2.4 – время выполнения второго запроса с индексом

### Практическое задание №3

### **Задание:**

- Подготовить 3-4 выборки, которые имеют осмысленное значение для предметной области, и также составить для них SQL-скрипты.
- Сформулировать 3-4 запроса на изменение и удаление из базы данных. Запросы должны быть сформулированы в терминах предметной области. Среди запросов обязательно должны быть такие, которые будут вызывать срабатывание ограничений целостности. Составить SQL-скрипты для выполнения этих запросов.

### **Выборка 1:**

```
select * from project
```

```
where project_complexity between '40' and '80'
```

```
order by deadline;
```

```
1 select * from project
2 where project_complexity between '40' and '80'
3 order by deadline;
```

The screenshot shows a database interface with a code editor at the top containing the SQL query. Below the code is a table with 7 rows of data. The table has five columns: project\_no, project\_name, project\_complexity, and deadline. The data is as follows:

	project_no [PK] integer	project_name	project_complexity double precision	deadline date
1	9	Иллюзия обмана	52	2004-02-23
2	10	Престиж	46	2004-04-15
3	7	Настоящий детектив	69	2011-10-30
4	4	Игра в имитацию	76	2012-11-07
5	1	Бойцовский клуб	60	2023-06-01
6	2	Барби	80	2046-04-12
7	3	Большая игра	45	2050-03-13

Рисунок 3.1 – результат выборки 1

## Выборка 2:

```
select * from assignment  
where handout_date between '2018-01-01' and '2019-01-01'  
order by assignment_complexity limit 100;
```

The screenshot shows a database interface with two main sections: a query editor at the top and a results viewer below it.

**Query Editor (Top):**

```
Запрос История запросов Scratch Pad
1 select * from assignment
2 where handout_date between '2018-01-01' and '2019-01-01'
3 order by assignment_complexity limit 100;
```

**Results Viewer (Bottom):**

Data Output Сообщения Notifications

	assignment_id [PK] integer	project_no integer	worker_id integer	handout_date date	planned_end_date date	real_end_date date	assignment_complexity double precision
1	495	6	9	2018-05-11	2022-07-18	2025-10-06	1
2	176	6	1	2018-07-26	2020-06-18	2023-11-06	1
3	604	2	6	2018-07-31	2021-05-25	2025-03-25	1
4	543	4	4	2018-11-14	2021-05-08	2025-06-20	1
5	116	9	4	2018-03-19	2022-09-18	2025-12-07	2
6	226	9	8	2018-06-29	2020-03-04	2025-02-15	2
7	57	5	9	2018-03-17	2020-09-04	2024-09-26	2
8	611	7	8	2018-05-28	2020-07-12	2025-06-06	2

Рисунок 3.2 – результат выборки 2

## Выборка 3:

```
select * from assignment  
where planned_end_date between '2020-01-01' and '2020-02-02'  
and worker_id=2  
order by assignment_complexity limit 100;
```

Запрос История запросов

```

1 select * from assignment
2 where planned_end_date between '2020-01-01' and '2020-02-02'
3 and worker_id=2
4 order by assignment_complexity limit 100;

```

Data Output Сообщения Notifications

	assignment_id [PK] integer	project_no integer	worker_id integer	handout_date date	planned_end_date date	real_end_date date	assignment_complexity double precision
1	345	6	2	2020-08-27	2020-01-22	2023-10-26	23
2	693	2	2	2020-03-18	2020-01-20	2023-07-10	50

Рисунок 3.3 – результат выборки 3

### Запрос 1:

```

insert into assignment
values
('1200', '11', '2', '2020-02-02', '2023-12-21', '2024-01-01', '30');

```

Запрос История запросов

```

1 insert into assignment
2 values
3 ('1200', '11', '2', '2020-02-02', '2023-12-21', '2024-01-01', '30');

```

Data Output Сообщения Notifications

ERROR: Ключ (project\_no)=(11) отсутствует в таблице "project".INSERT или UPDATE в таблице "assignment" нарушает ограничение внешнего ключа "assignment\_project\_no\_fkey"

ОШИБКА: INSERT или UPDATE в таблице "assignment" нарушает ограничение внешнего ключа "assignment\_project\_no\_fkey"  
SQL-состояние: 23503  
Подробности: Ключ (project\_no)=(11) отсутствует в таблице "project".

Рисунок 3.4 – результат запроса 1

### Исправленный запрос:

```

insert into assignment
values

```

```
('1200', '10', '2', '2020-02-02', '2023-12-21', '2024-01-01', '30');
```

The screenshot shows a database query editor window. At the top, there are tabs for 'Запрос' (Query) and 'История запросов' (Query History). Below the tabs is a code editor containing the following SQL statement:

```
1 insert into assignment
2 values
3 ('1200', '10', '2', '2020-02-02', '2023-12-21', '2024-01-01', '30');
```

Below the code editor, there are three tabs: 'Data Output', 'Сообщения' (Messages), and 'Notifications'. The 'Сообщения' tab is selected, showing the message 'INSERT 0 1'. Underneath this, the message 'Запрос завершён успешно, время выполнения: 65 msec.' is displayed.

Рисунок 3.5 – результат исправленного запроса

### Запрос 2:

```
DELETE
```

```
FROM assignment
```

```
WHERE assignment_id = 1200;
```

The screenshot shows a database query editor window. At the top, there are tabs for 'Запрос' (Query) and 'История запросов' (Query History). Below the tabs is a code editor containing the following SQL statement:

```
1 DELETE
2 FROM assignment
3 WHERE assignment_id = 1200;
4
```

Below the code editor, there are three tabs: 'Data Output', 'Сообщения' (Messages), and 'Notifications'. The 'Сообщения' tab is selected, showing the message 'DELETE 1'. Underneath this, the message 'Запрос завершён успешно, время выполнения: 52 msec.' is displayed.

Рисунок 3.6 – результат запроса 2

### Запрос 3:

```
update assignment
```

```
set assignment_complexity=99
```

```
where worker_id=2;
```

The screenshot shows a database interface with a query editor and a results pane. The query editor contains the following SQL code:

```
1 update assignment
2 set assignment_complexity=99
3 where worker_id=2;
4
```

The results pane shows the output of the query:

UPDATE	85
Сообщения	

Рисунок 3.7 – результат запроса 3

#### Практическое задание №4

##### **Задание:**

- Составить SQL-скрипты для создания нескольких представлений, которые позволяли бы упростить манипуляции с данными или позволяли бы ограничить доступ к данным, предоставляя только необходимую информацию.
- Продемонстрировать изменение и вставку данных через представления.
- Продемонстрировать невозможность изменения данных через представление.
- Продемонстрировать полезность материализованного представления.

##### **Представление 1:**

*SQL-скрипт:*

```
CREATE OR REPLACE VIEW count_worker_and_project AS
SELECT project_no, worker_id, COUNT(*)
FROM assignment
GROUP BY worker_id, project_no
ORDER BY worker_id, project_no;
```

```
Запрос История запросов
1 CREATE OR REPLACE VIEW count_worker_and_project AS
2 SELECT project_no, worker_id, COUNT(*)
3 FROM assignment
4 GROUP BY worker_id, project_no
5 ORDER BY worker_id, project_no;
6
```

Data Output Сообщения Notifications

CREATE VIEW

Запрос завершён успешно, время выполнения: 56 msec.

Рисунок 4.1 – результат выполнения

*Невозможность изменения данных через представление:*

```
Запрос История запросов Scratch Pad
1 INSERT INTO count_worker_and_project
2 VALUES
3 ('1', '1', '10');
4
```

Data Output Сообщения Notifications

ERROR: Представления с GROUP BY не обновляются автоматически. Вставить данные в представление "count\_worker\_and\_project" нельзя

ОШИБКА: вставить данные в представление "count\_worker\_and\_project" нельзя  
SQL-состояние: 55000  
Подробности: Представления с GROUP BY не обновляются автоматически.  
Подсказка: Чтобы представление допускало добавление данных, установите триггер INSTEAD OF INSERT или безусловное правило ON INSERT DO INSTEAD.

Рисунок 4.2 – результат выполнения

**Представление 2:**

*SQL-скрипт:*

drop view if exists tr;

create or replace view vtoroe as

```
select * from assignment where handout_date>'2019-09-09';
```

*Изменение и вставка данных через представление:*

update vtoroe

```
set handout_date='2020-12-12' where worker_id='2';
```

The screenshot shows a database interface with two main sections: a query editor and a data output viewer.

**Query Editor (Top):**

```
1 update vtoroe
2 set handout_date='2020-12-12' where worker_id='2';
3 select * from vtoroe where worker_id='2';
4
```

**Data Output (Bottom):**

	assignment_id integer	project_no integer	worker_id integer	handout_date date	planned_end_date date	real_end_date date	assignment_complexity double precision
1	11	7	2	2020-12-12	2020-03-23	2023-10-30	99
2	25	7	2	2020-12-12	2022-02-19	2023-11-01	99
3	44	9	2	2020-12-12	2021-08-24	2024-01-18	99
4	74	5	2	2020-12-12	2022-02-26	2025-12-23	99
5	88	2	2	2020-12-12	2022-03-10	2023-10-14	99

Рисунок 4.3 – измененное представление

*Вставка данных через представление:*

insert into vtoroe

values

```
('1200', '3', '2', '2020-02-02', '2024-12-12', '2025-05-05', '60')
```

The screenshot shows a database interface with two main sections: a query editor and a data output viewer.

**Query Editor (Top):**

```
1 insert into vtoroe
2 values
3 ('1200', '3', '2', '2020-02-02', '2024-12-12', '2025-05-05', '60')
```

**Data Output (Bottom):**

	assignment_id integer	project_no integer	worker_id integer	handout_date date	planned_end_date date	real_end_date date	assignment_complexity double precision
1	1200	3	2	2020-02-02	2024-12-12	2025-05-05	60

Messages tab (bottom):

```
INSERT 0 1
```

Success message (bottom):

```
Запрос завершён успешно, время выполнения: 57 msec.
```

Рисунок 4.4 – результат выполнения

### **Практическое задание №5**

#### **Задание:**

Необходимо подготовить SQL-скрипты для проверки наличия аномалий (потерянных изменений, грязных чтений, неповторяющихся чтений, фантомов) при параллельном исполнении транзакций на различных уровнях изолированности SQL/92 (READ UNCOMMITTED, READ COMMITTED, REPEATABLE READ, SERIALIZABLE). Подготовленные скрипты должны работать с одной из таблиц, созданных в практическом задании №2.1. Для проверки наличия аномалий потребуются два параллельных сеанса, операторы в которых выполняются пошагово:

- Установить в обоих сеансах уровень изоляции READ UNCOMMITTED.
- Выполнить сценарии проверки наличия аномалий потерянных изменений и грязных чтений.
- Установить в обоих сеансах уровень изоляции READ COMMITTED.
- Выполнить сценарии проверки наличия аномалий грязных чтений и неповторяющихся чтений.
- Установить в обоих сеансах уровень изоляции REPEATABLE READ.  
Выполнить сценарии проверки наличия аномалий неповторяющихся чтений и фантомов.
- Установить в обоих сеансах уровень изоляции SERIALIZABLE.

**Уровень изоляции READ UNCOMMITTED**

```

lab2=# begin;
BEGIN
lab2=# set transaction isolation level read uncommitted;
SET
lab2=# show transaction_isolation;
transaction_isolation
-----
read uncommitted
(1 行)

lab2=# update assignment
lab2-*# set assignment_complexity=assignment_complexity+10
lab2-*# where assignment_complexity<89 and assignment_id='12';
UPDATE 1
lab2=*# select * from assignment where assignment_id='12';
 assignment_id | project_no | worker_id | handout_date | planned_end_date | real_end_date | assignment_complexity
-----+-----+-----+-----+-----+-----+-----+
      12 |         1 |         9 | 2020-01-12 | 2021-02-18 | 2024-09-14 |             98
(1 行)

lab2=*#

```

Рисунок 5.1 – Транзакция 1 (Терминал №1)

```

lab2=# begin;
BEGIN
lab2=# set transaction isolation level read uncommitted;
SET
lab2=*# select * from assignment where assignment_id='12';
 assignment_id | project_no | worker_id | handout_date | planned_end_date | real_end_date | assignment_complexity
-----+-----+-----+-----+-----+-----+-----+
      12 |         1 |         9 | 2020-01-12 | 2021-02-18 | 2024-09-14 |             88
(1 行)


```

Рисунок 5.2 – Транзакция 1 (Терминал №2)

Вторая транзакция не видит изменение значения атрибута `assignment_complexity`, произведенное в первой — незафиксированной — транзакции.

Чтение грязных данных не произошло, так как в PostgreSQL требования, предъявляемые к этому уровню, более строгие, чем в стандарте

**Уровень изоляции **READ COMMITTED****

```

lab2=# begin isolation level read committed;
BEGIN
lab2=*# show transaction_isolation;
transaction_isolation
-----
read_committed
(1 є€‰юрп)

lab2=*# update assignment
lab2-*# set assignment_complexity=assignment_complexity-10
lab2-*# where assignment_id='12';
UPDATE 1
lab2=*# select * from assignment where assignment_id='12';
assignment_id | project_no | worker_id | handout_date | planned_end_date | real_end_date | assignment_complexity
-----+-----+-----+-----+-----+-----+-----+
      12 |         1 |         9 | 2020-01-12 | 2021-02-18 | 2024-09-14 |          78
(1 є€‰юрп)

```

Рисунок 5.3 – Транзакция 2 (Терминал №1)

```

lab2=# begin;
BEGIN
lab2=*# update assignment
lab2-*# set assignment_complexity=assignment_complexity-20
lab2-*# where assignment_id='12';
|

```

Рисунок 5.4 – Транзакция 2 (Терминал №2)

Команда UPDATE в первой транзакции заблокировала строку в таблице assignment\_tmp. После выполнения команды ROLLBACK (отмена изменений) в первом терминале, команда UPDATE во втором терминале завершилась. Таким образом, грязных чтений не производится, в отличие от неповторяющегося чтения.

```

lab2=# begin;
BEGIN
lab2=*# update assignment
lab2-*# set assignment_complexity=assignment_complexity-20
lab2-*# where assignment_id='12';
UPDATE 1
lab2=*#

```

Рисунок 5.5 – транзакция 2 (Терминал №2) после завершения транзакции в первом терминале

## Уровень изоляции REPEATABLE READ

```
lab2=# BEGIN TRANSACTION ISOLATION LEVEL REPEATABLE READ;
BEGIN
lab2=*# select * from assignment limit 10;
assignment_id | project_no | worker_id | handout_date | planned_end_date | real_end_date | assignment_complexity
-----+-----+-----+-----+-----+-----+-----+
1 | 7 | 5 | 2018-11-29 | 2020-06-23 | 2024-07-02 | 8
2 | 9 | 9 | 2019-01-24 | 2020-10-08 | 2023-10-28 | 72
3 | 10 | 6 | 2019-05-16 | 2020-01-08 | 2025-12-10 | 41
4 | 10 | 1 | 2018-12-08 | 2020-03-02 | 2023-04-05 | 64
5 | 1 | 3 | 2019-11-29 | 2022-02-27 | 2025-07-06 | 22
6 | 6 | 1 | 2018-06-19 | 2021-07-17 | 2023-04-10 | 77
7 | 5 | 9 | 2018-03-13 | 2022-07-21 | 2024-04-17 | 33
8 | 3 | 3 | 2019-06-28 | 2020-06-01 | 2023-07-06 | 81
9 | 3 | 1 | 2018-05-06 | 2021-04-05 | 2025-09-13 | 20
10 | 3 | 4 | 2019-06-23 | 2020-11-22 | 2025-02-20 | 1
(10 rows)
```

Рисунок 5.6 – транзакция 3 (Терминал №1)

```
lab2=# BEGIN TRANSACTION ISOLATION LEVEL REPEATABLE READ;
BEGIN
lab2=*# update assignment
lab2-*# set assignment_complexity=assignment_complexity+10
lab2-*# where assignment_id='1';
UPDATE 1
lab2=*# end;
COMMIT
```

Рисунок 5.7 – транзакция 3 (Терминал №2)

```
lab2=*# select * from assignment limit 10;
assignment_id | project_no | worker_id | handout_date | planned_end_date | real_end_date | assignment_complexity
-----+-----+-----+-----+-----+-----+-----+
1 | 7 | 5 | 2018-11-29 | 2020-06-23 | 2024-07-02 | 8
2 | 9 | 9 | 2019-01-24 | 2020-10-08 | 2023-10-28 | 72
3 | 10 | 6 | 2019-05-16 | 2020-01-08 | 2025-12-10 | 41
4 | 10 | 1 | 2018-12-08 | 2020-03-02 | 2023-04-05 | 64
5 | 1 | 3 | 2019-11-29 | 2022-02-27 | 2025-07-06 | 22
6 | 6 | 1 | 2018-06-19 | 2021-07-17 | 2023-04-10 | 77
7 | 5 | 9 | 2018-03-13 | 2022-07-21 | 2024-04-17 | 33
8 | 3 | 3 | 2019-06-28 | 2020-06-01 | 2023-07-06 | 81
9 | 3 | 1 | 2018-05-06 | 2021-04-05 | 2025-09-13 | 20
10 | 3 | 4 | 2019-06-23 | 2020-11-22 | 2025-02-20 | 1
(10 rows)
```

Рисунок 5.8 – вывод таблицы в первом терминале сразу после изменения строки  
во втором терминале

Изменения во втором терминале не повлияли на выводные данные в первом терминале, так как первый терминал всё ещё использует снимок таблицы, сделанный до преобразований.

```

lab2=# BEGIN TRANSACTION ISOLATION LEVEL REPEATABLE READ;
BEGIN
lab2=**# select * from assignment limit 10;
assignment_id | project_no | worker_id | handout_date | planned_end_date | real_end_date | assignment_complexity
-----+-----+-----+-----+-----+-----+-----
2 | 9 | 9 | 2019-01-24 | 2020-10-08 | 2023-10-28 | 72
3 | 10 | 6 | 2019-05-16 | 2020-01-08 | 2025-12-10 | 41
4 | 10 | 1 | 2018-12-08 | 2020-03-02 | 2023-04-05 | 64
5 | 1 | 3 | 2019-11-29 | 2022-02-27 | 2025-07-06 | 22
6 | 6 | 1 | 2018-06-19 | 2021-07-17 | 2023-04-10 | 77
7 | 5 | 9 | 2018-03-13 | 2022-07-21 | 2024-04-17 | 33
8 | 3 | 3 | 2019-06-28 | 2020-06-01 | 2023-07-06 | 81
9 | 3 | 1 | 2018-05-06 | 2021-04-05 | 2025-09-13 | 20
10 | 3 | 4 | 2019-06-23 | 2020-11-22 | 2025-02-20 | 1
1 | 7 | 5 | 2018-11-29 | 2020-06-23 | 2024-07-02 | 18
(10 ёсёлю)

```

Рисунок 5.9 – вывод таблицы в первом терминале после завершения предыдущей транзакции

Таким образом, феномена неповторяющегося чтения данных не производились. Однако в PostgreSQL на этом уровне не допускается и чтение фантомных строк.

### Уровень изоляции SERIALIZABLE

```

lab2=# begin transaction isolation level serializable;
BEGIN
lab2=**# update assignment
lab2=**# set assignment_complexity=assignment_complexity+10
lab2=**# where assignment_complexity='62';
UPDATE 13
lab2=**# select * from assignment order by assignment_id limit 10;
assignment_id | project_no | worker_id | handout_date | planned_end_date | real_end_date | assignment_complexity
-----+-----+-----+-----+-----+-----+-----
1 | 7 | 5 | 2018-11-29 | 2020-06-23 | 2024-07-02 | 72
2 | 9 | 9 | 2019-01-24 | 2020-10-08 | 2023-10-28 | 72
3 | 10 | 6 | 2019-05-16 | 2020-01-08 | 2025-12-10 | 41
4 | 10 | 1 | 2018-12-08 | 2020-03-02 | 2023-04-05 | 64
5 | 1 | 3 | 2019-11-29 | 2022-02-27 | 2025-07-06 | 22
6 | 6 | 1 | 2018-06-19 | 2021-07-17 | 2023-04-10 | 77
7 | 5 | 9 | 2018-03-13 | 2022-07-21 | 2024-04-17 | 33
8 | 3 | 3 | 2019-06-28 | 2020-06-01 | 2023-07-06 | 81
9 | 3 | 1 | 2018-05-06 | 2021-04-05 | 2025-09-13 | 20
10 | 3 | 4 | 2019-06-23 | 2020-11-22 | 2025-02-20 | 1
(10 ёсёлю)

```

Рисунок 5.10 – Транзакция 4 (Терминал №1)

```

lab2=# begin transaction isolation level serializable;
BEGIN
lab2=# update assignment
lab2-*# set assignment_complexity=assignment_complexity+10
lab2-*# where assignment_complexity='72';
UPDATE 6
lab2=# select * from assignment order by assignment_id limit 10;
assignment_id | project_no | worker_id | handout_date | planned_end_date | real_end_date | assignment_complexity
-----+-----+-----+-----+-----+-----+-----
      1 |      7 |      5 | 2018-11-29 | 2020-06-23 | 2024-07-02 |          62
      2 |      9 |      9 | 2019-01-24 | 2020-10-08 | 2023-10-28 |          82
      3 |     10 |      6 | 2019-05-16 | 2020-01-08 | 2025-12-10 |          41
      4 |     10 |      1 | 2018-12-08 | 2020-03-02 | 2023-04-05 |          64
      5 |      1 |      3 | 2019-11-29 | 2022-02-27 | 2025-07-06 |          22
      6 |      6 |      1 | 2018-06-19 | 2021-07-17 | 2023-04-10 |          77
      7 |      5 |      9 | 2018-03-13 | 2022-07-21 | 2024-04-17 |          33
      8 |      3 |      3 | 2019-06-28 | 2020-06-01 | 2023-07-06 |          81
      9 |      3 |      1 | 2018-05-06 | 2021-04-05 | 2025-09-13 |          20
     10 |      3 |      4 | 2019-06-23 | 2020-11-22 | 2025-02-20 |          1
(10 rows)

```

Рисунок 5.11 – Транзакция 4 (Терминал №2)

Изменение, произведенное в первой транзакции, вторая транзакция не видит, поскольку на уровне изоляции SERIALIZABLE каждая транзакция работает с тем снимком базы данных, которых был сделан в ее начале, т. е. непосредственно перед выполнением ее первого оператора. Поэтому обновляется только одна строка.

**Вывод:** мы ознакомились с операторами создания БД, изучили типы данных, научились добавлять записи в таблицы, удалять и изменять их, ознакомились с механизмами контроля согласования БД, транзакциями и триггерами.