

**ТЕМА: ТРАНЗАКЦИИ И УПРАВЛЕНИЕ  
КОНКУРЕНТНЫМ ДОСТУПОМ**

ДИСЦИПЛИНА: ВВЕДЕНИЕ В БАЗЫ ДАННЫХ

ДЛЯ СТУДЕНТОВ ОБРАЗОВАТЕЛЬНОЙ ПРОГРАММЫ  
6B06105 «DATA SCIENCE»

СТАРШИЙ ПРЕПОДАВАТЕЛЬ КЛЮЕВА Е.Г.

# План лекции:

1. Понятие транзакции. Свойства транзакций;

2. Модели транзакций;

3. Журнализация изменений БД;

4. Проблемы параллелизма транзакций.

# ПОНЯТИЕ ТРАНЗАКЦИИ

*Транзакция* — это последовательность операторов манипулирования данными, выполняющаяся как единое целое и переводящая базу данных из одного целостного состояния в другое.



# СВОЙСТВА ТРАНЗАКЦИЙ

Традиционные транзакции обладают четырьмя свойствами, известными как ACID:

- ✓ атомарность (Atomicity);
- ✓ согласованность (Consistency);
- ✓ изолированность (Isolation);
- ✓ долговечность (Durability).



# УПРАВЛЕНИЕ ТРАНЗАКЦИЯМИ

Под *управлением транзакциями* понимается способность управлять различными операциями над данными, которые выполняются внутри реляционной СУБД.



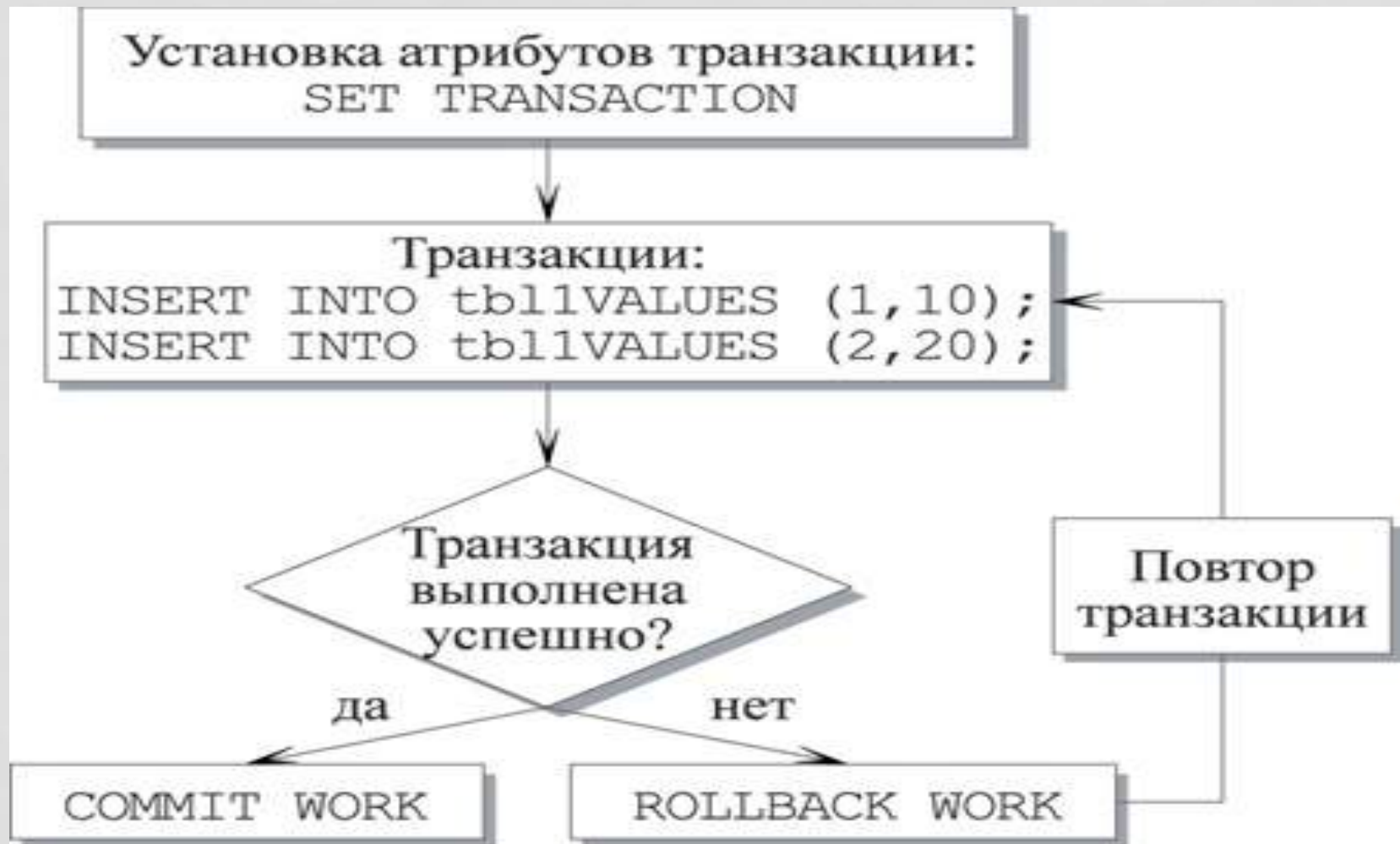
# УПРАВЛЕНИЕ ТРАНЗАКЦИЯМИ

Существует два способа (две команды) завершения транзакций:

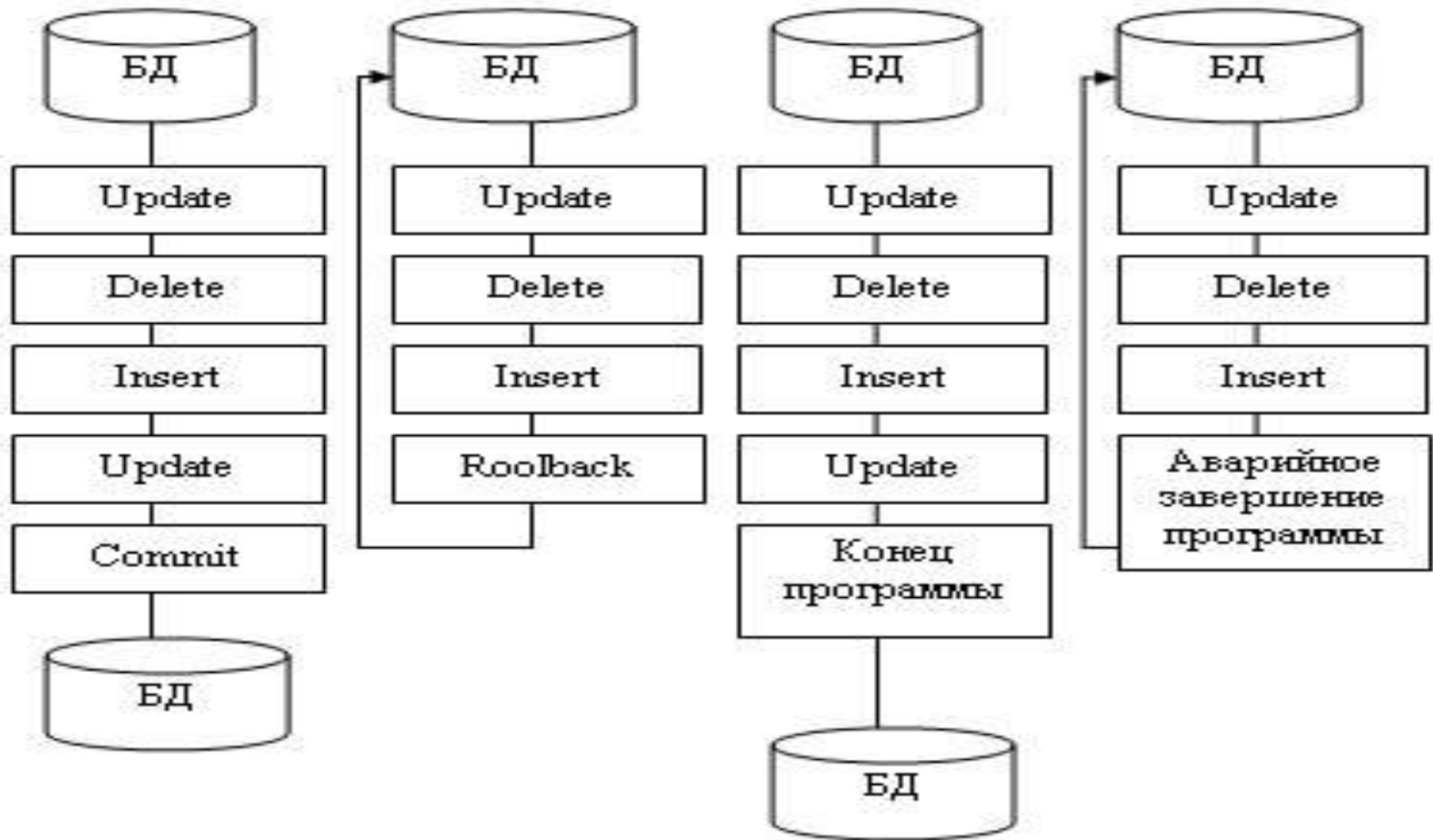
- ✓ фиксация (COMMIT);
- ✓ откат (ROLLBACK).



# УПРАВЛЕНИЕ ТРАНЗАКЦИЯМИ



# МОДЕЛЬ ТРАНЗАКЦИИ ANSI SQL





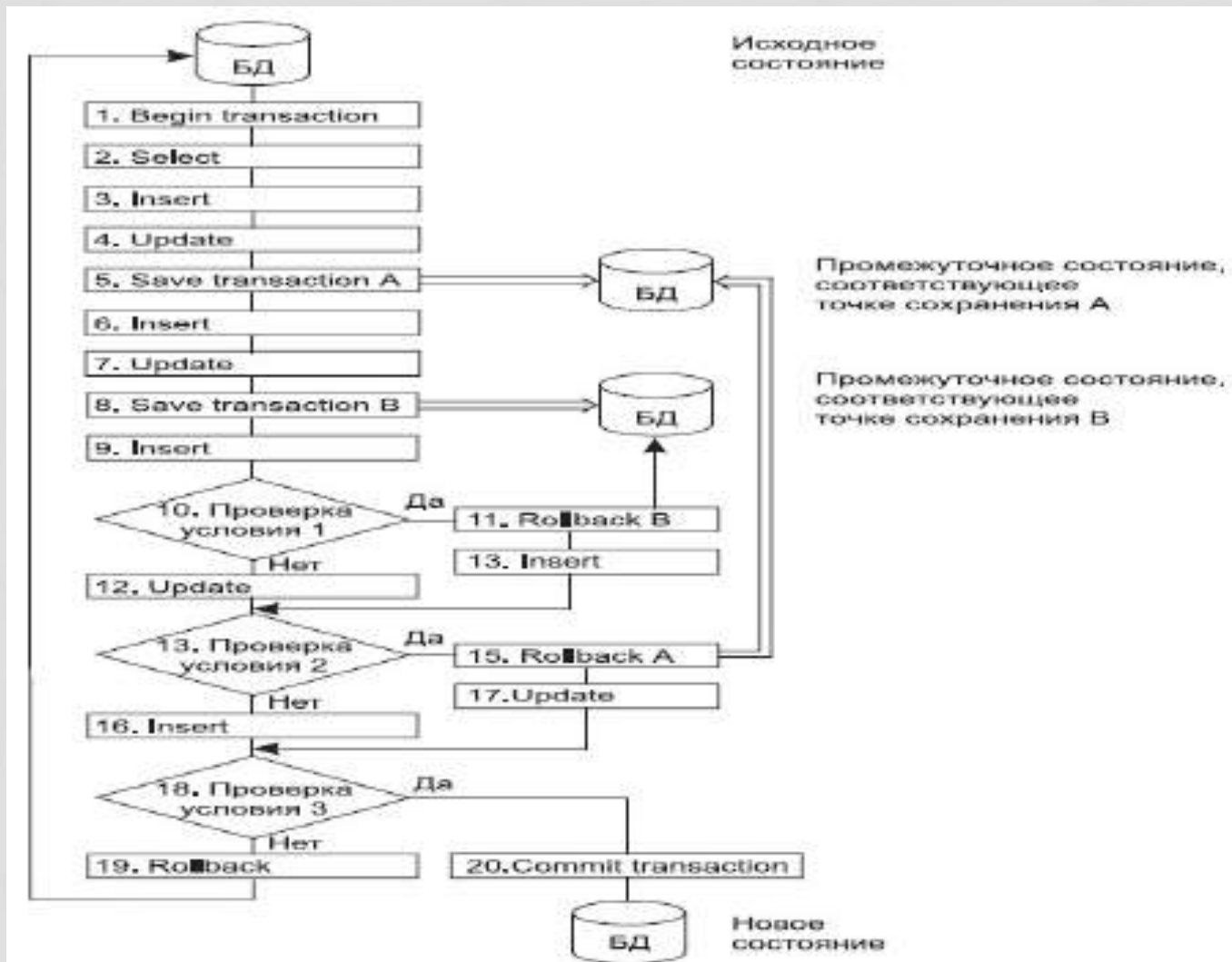
# РАСШИРЕННАЯ МОДЕЛЬ ТРАНЗАКЦИЙ

В расширенной модели транзакций используются следующие четыре оператора:

- ✓ BEGIN TRANSACTION;
- ✓ COMMIT TRANSACTION;
- ✓ SAVE TRANSACTION;
- ✓ ROLLBACK и ROLLBACK B.



# РАСШИРЕННАЯ МОДЕЛЬ ТРАНЗАКЦИЙ



# ЖУРНАЛИЗАЦИЯ ИЗМЕНЕНИЙ БД

Общей целью журнализации изменений БД является обеспечение возможности восстановления согласованного состояния базы данных после любого сбоя.



# ЖУРНАЛИЗАЦИЯ ИЗМЕНЕНИЙ БД

Общие принципы восстановления данных:

- результаты зафиксированных транзакций должны быть сохранены в восстановленном состоянии базы данных;
- результаты незафиксированных транзакций должны отсутствовать в восстановленном состоянии базы данных.

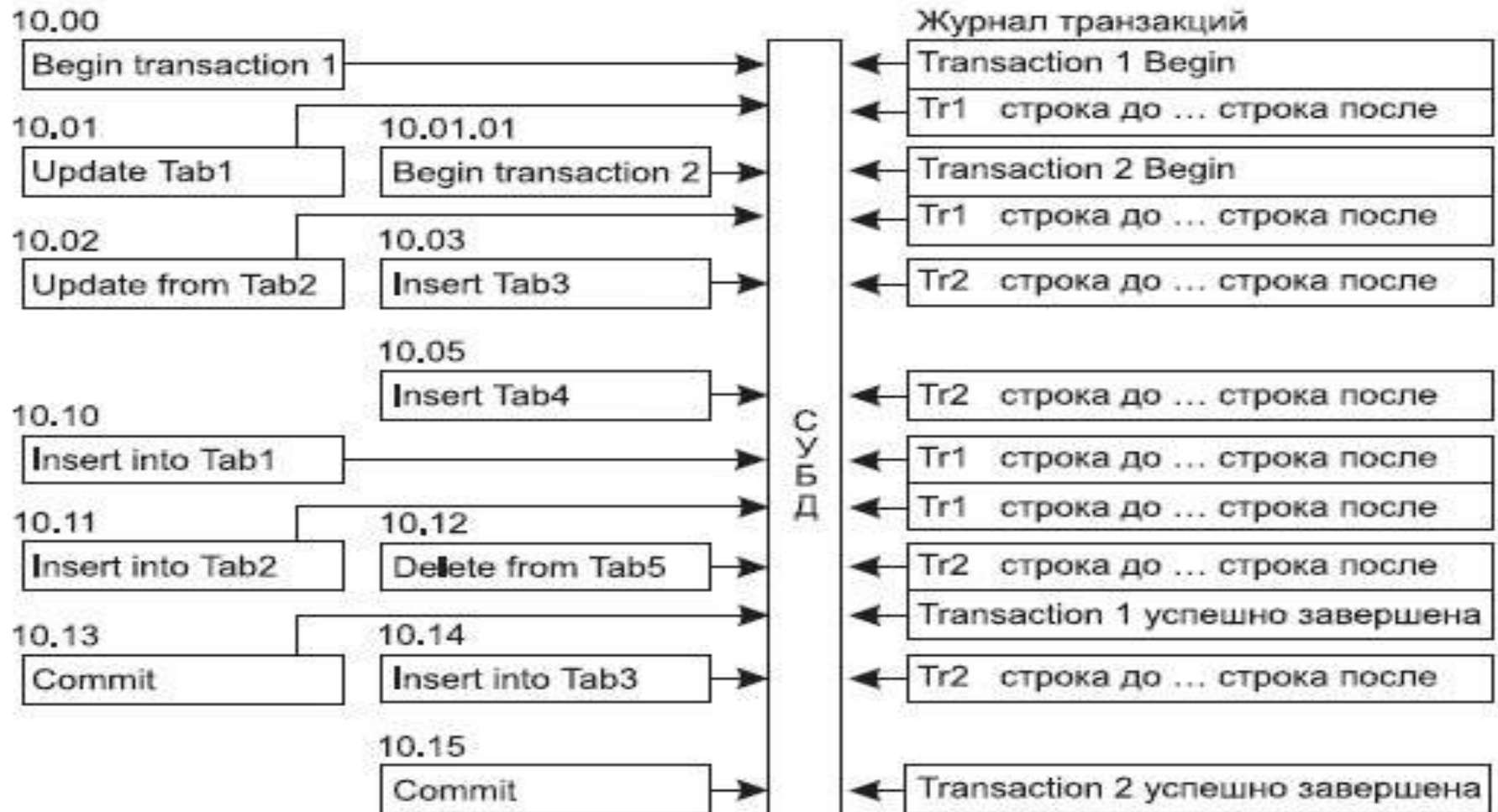
# ЖУРНАЛИЗАЦИЯ ИЗМЕНЕНИЙ БД

Варианты ведения журнальной информации в СУБД:

1. для каждой транзакции поддерживается отдельный локальный журнал изменений БД этой транзакцией и поддерживается общий журнал изменений БД;

2. поддерживается только общий журнал изменений БД.

# СТРУКТУРА ЖУРНАЛА ТРАНЗАКЦИЙ



# ЖУРНАЛ ТРАНЗАКЦИЙ

Имеются два альтернативных варианта ведения журнала транзакций:

- протокол с отложенными обновлениями;
- протокол с немедленными обновлениями.



# ПРОТОКОЛ С ОТЛОЖЕННЫМИ ОБНОВЛЕНИЯМИ

<T1 Begin transaction>

<T1, ID\_RECORD, атрибут, новое значение ... >

.....

<T1 COMMIT>





# ПРОТОКОЛ С НЕМЕДЛЕННЫМИ ОБНОВЛЕНИЯМИ

<T1 Begin transaction>

<T1, ID\_RECORD, атрибут новое значение  
старое значение ...>

.....

<T1 COMMIT>



# ПРОТОКОЛ С НЕМЕДЛЕННЫМИ ОБНОВЛЕНИЯМИ

<T1 Begin transaction>

<T1, ID\_RECORD, атрибут новое значение  
старое значение ...>

.....

UNDO()



# ЖУРНАЛИЗАЦИЯ ИЗМЕНЕНИЙ БД

Ситуации, при которых требуется производить восстановление состояния БД по журналу:

- явное завершение транзакции оператором ROLLBACK;
- мягкий сбой;
- жесткий сбой.



# ЖУРНАЛИЗАЦИЯ ИЗМЕНЕНИЙ БД

Протокол журнализации (и управления буферизацией) называется **Write Ahead Log (WAL)** - "пиши сначала в журнал", и состоит в том, что если требуется вытолкнуть во внешнюю память измененный объект базы данных, то перед этим нужно гарантировать выталкивание во внешнюю память журнала записи о его изменении.

# ИНДИВИДУАЛЬНЫЙ ОТКАТ ТРАНЗАКЦИИ

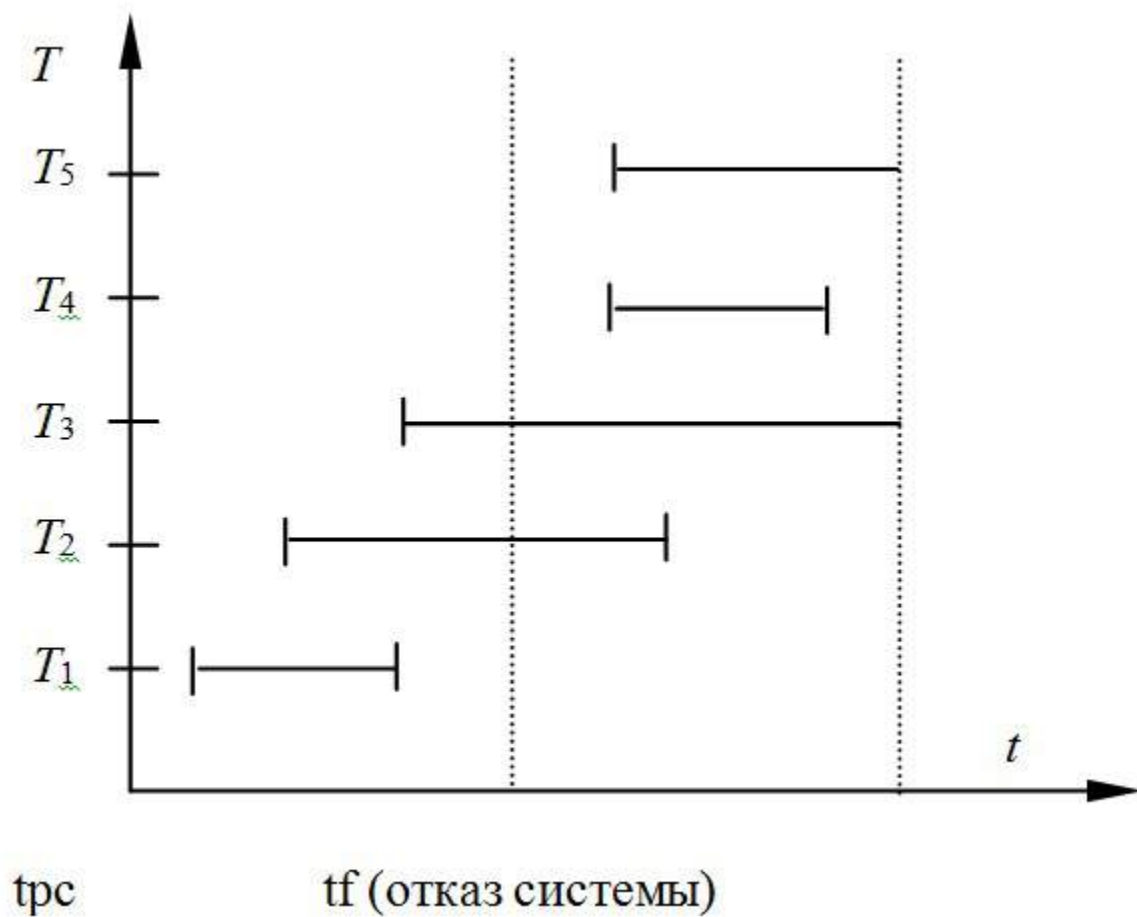
- Выбирается очередная запись из списка данной транзакции.
- Выполняется противоположная по смыслу операция.
- Любая из этих обратных операций также журнализируется.
- При успешном завершении отката в журнал заносится запись о конце транзакции. С точки зрения журнала такая транзакция является зафиксированной.

# ВОССТАНОВЛЕНИЕ БД ПОСЛЕ МЯГКОГО СБОЯ

Состояние внешней памяти базы данных называется физически согласованным, если наборы страниц всех объектов согласованы, т.е. соответствуют состоянию объекта либо после его изменения, либо до изменения.



# ВОССТАНОВЛЕНИЕ БД ПОСЛЕ МЯГКОГО СБОЯ



# ВОССТАНОВЛЕНИЕ БД ПОСЛЕ МЯГКОГО СБОЯ

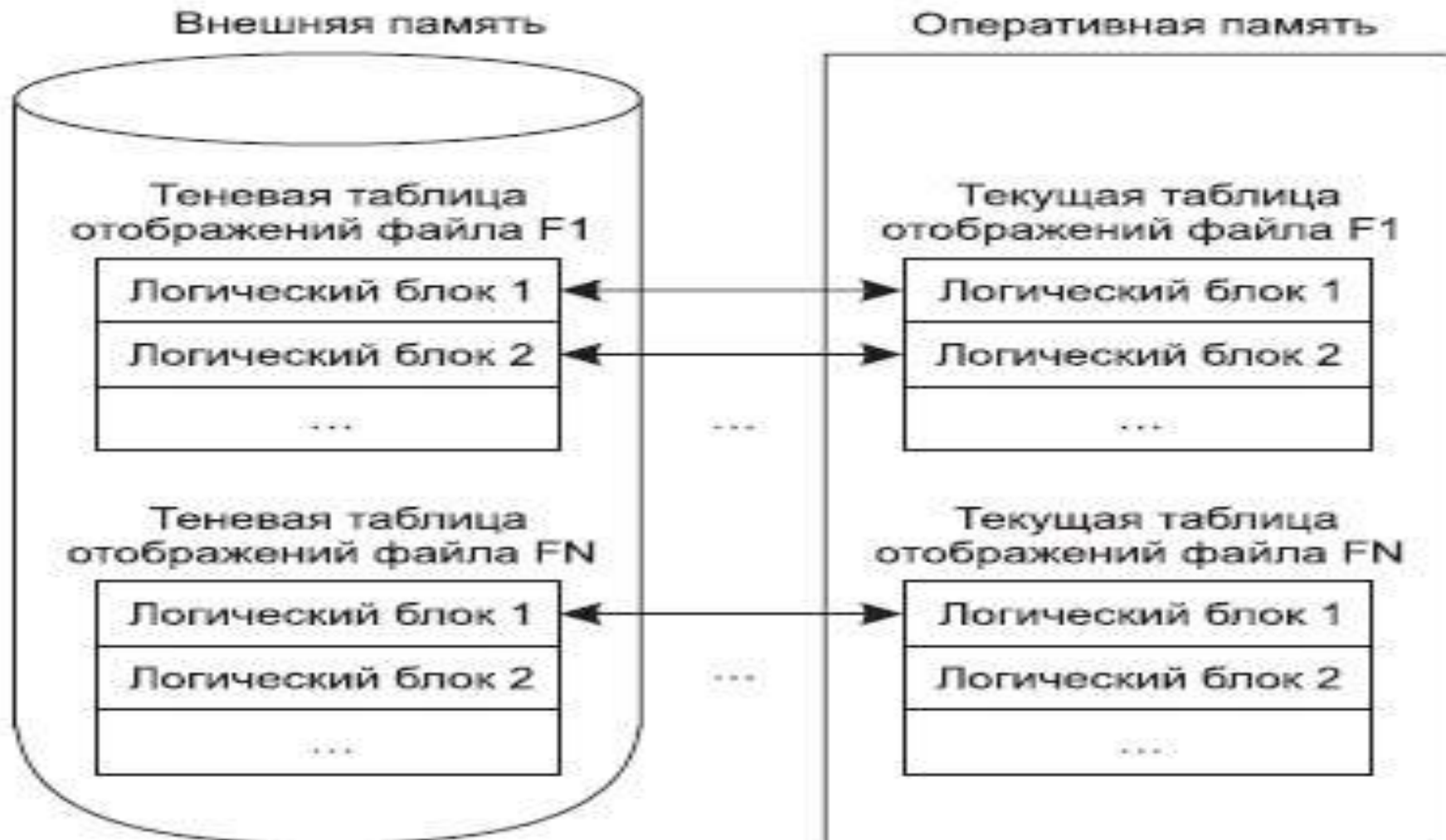
Основные подходы к восстановлению  
состояние базы данных в момент `trс`:

- подход, основанный на использовании  
теневого механизма;
- подход, в котором применяется  
журнализация страничных изменений базы  
данных.





# ТЕНЕВОЙ МЕХАНИЗМ



# ВОССТАНОВЛЕНИЕ БД ПОСЛЕ ЖЁСТКОГО СБОЯ

Основа восстановления БД после жёсткого сбоя: журнал и архивная копия базы данных.

При восстановлении происходит следующее:

- по журналу в прямом направлении выполняются все операции;
- для транзакций, которые не закончились к моменту сбоя, выполняется откат.

# СПОСОБЫ СОЗДАНИЯ АРХИВНЫХ КОПИЙ БАЗЫ ДАННЫХ

- Архивирование базы данных при переполнении журнала.
- Архивирование базы данных и самого журнала реже, чем он переполняется.



*freepik.com*

WiseGEEK

# ПРОБЛЕМЫ ПАРАЛЛЕЛИЗМА ТРАНЗАКЦИЙ

Термин *параллелизм* означает возможность одновременной обработки СУБД нескольких транзакций, запрашивающих одни и те же данные, причем в одно и то же время, а также рассматривает проблемы и методы обеспечения изолированности транзакций.

# ПРОБЛЕМЫ ПАРАЛЛЕЛИЗМА ТРАНЗАКЦИЙ

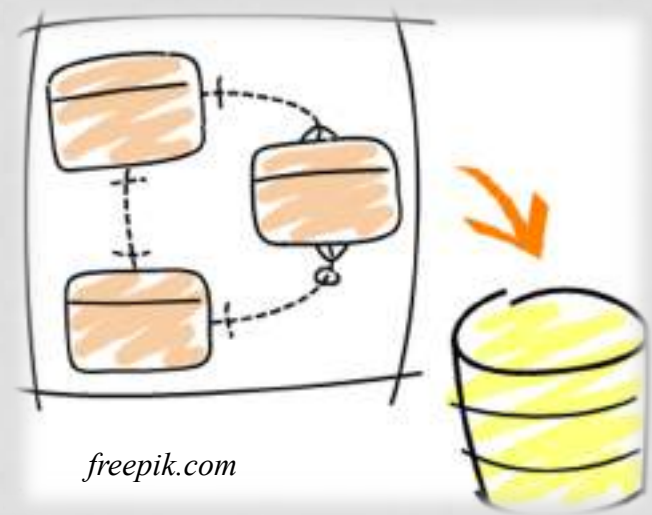
Существуют три основные проблемы параллелизма:

- потери результатов обновления (пропавшие изменения);
- незафиксированной зависимости (чтение "грязных данных");
- несовместимого анализа.

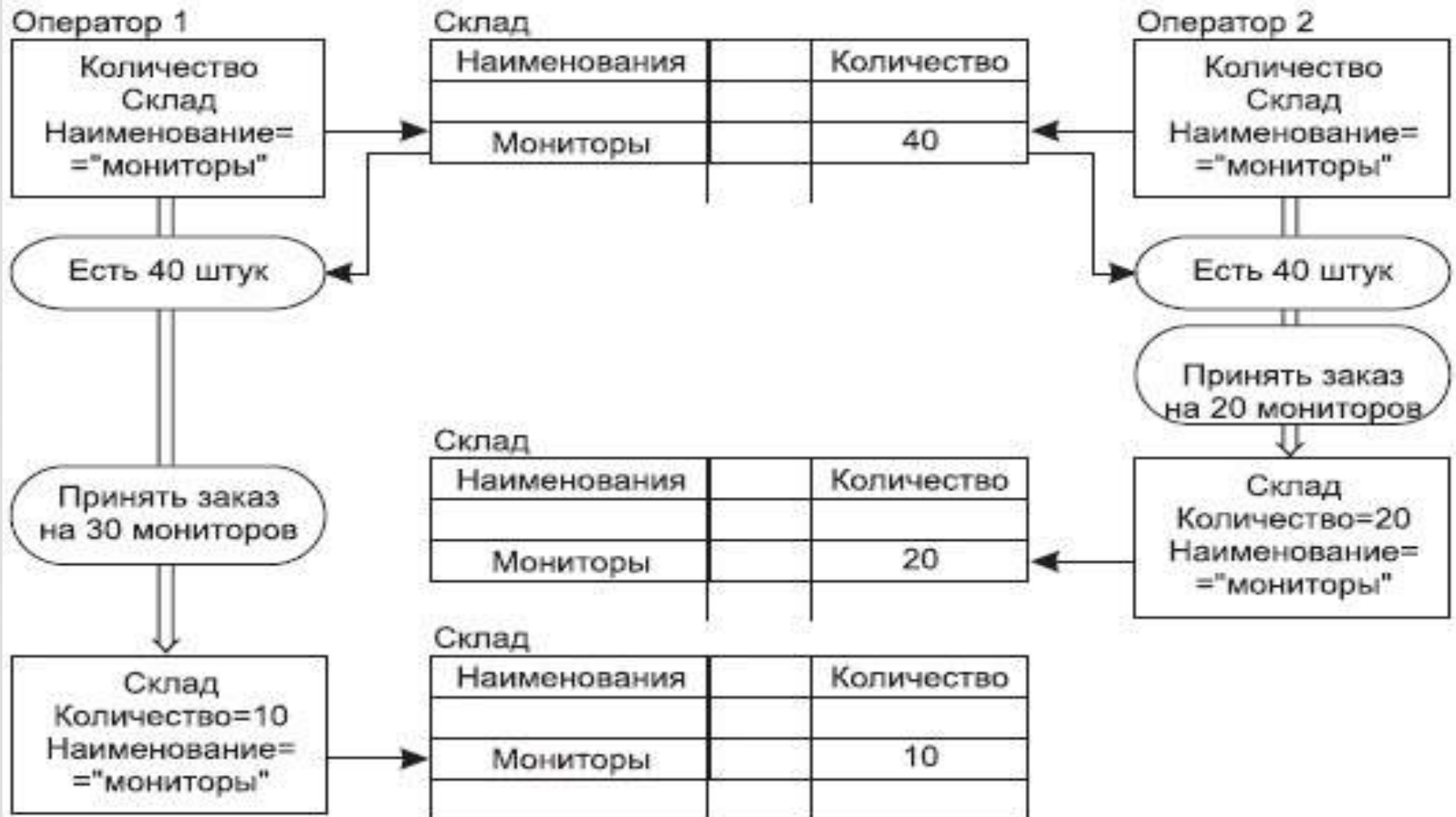


# ПРОБЛЕМЫ ПАРАЛЛЕЛИЗМА ТРАНЗАКЦИЙ

*Потеря результатов обновления возникает в ситуации, когда две транзакции одновременно читают и изменяют одну и ту же запись в БД. Причем одна транзакция записывает в строку БД свое значение, а вторая транзакция вносит в эту же строку БД свои изменения.*



# ПРОБЛЕМЫ ПАРАЛЛЕЛИЗМА ТРАНЗАКЦИЙ



# ПРОБЛЕМЫ ПАРАЛЛЕЛИЗМА ТРАНЗАКЦИЙ

Транзакция 1		Транзакция 2
SELECT f2 ( <i>f2=40</i> ) FROM tbl1 WHERE f1=1;	<i>t1</i>	
	<i>t2</i>	SELECT f2 ( <i>f2=40</i> ) FROM tbl1 WHERE f1=1;
UPDATE tbl1 SET f2=F2-30 ( <i>f2=10</i> ) WHERE f1=1;	<i>t3</i>	
	<i>t4</i>	UPDATE tbl1 SET SET f2=F2-20 ( <i>f2=20</i> ) WHERE f1=1
COMMIT	<i>t5</i>	—
—	<i>t6</i>	COMMIT



# ПРОБЛЕМЫ ПАРАЛЛЕЛИЗМА ТРАНЗАКЦИЙ

*Незафиксированная зависимость (чтение "грязных данных")* возникает, если одна транзакция выполняет чтение некоторой строки таблицы БД, которая в данный момент обновляется другой транзакцией, но это обновление еще не зафиксировано. Первая транзакция выполнила чтение «грязных» данных, которых нет и не было в базе данных в результате аннулирования изменения более поздним откатом второй транзакции.

# ПРОБЛЕМЫ ПАРАЛЛЕЛИЗМА ТРАНЗАКЦИЙ

Транзакция 1		Транзакция 2
SELECT f2 ( <i>f2=40</i> ) FROM tbl1 WHERE f1=1;	<i>t1</i>	
UPDATE tbl1 SET f2=f2-30 ( <i>f2=10</i> ) WHERE f1=1;	<i>t2</i>	
	<i>t3</i>	SELECT f2 ( <i>f2=10</i> ) FROM tbl1 WHERE f1=1;
	<i>t4</i>	COMMIT
ROLLBACK WORK;	<i>t5</i>	

# ПРОБЛЕМЫ ПАРАЛЛЕЛИЗМА ТРАНЗАКЦИЙ

Проблема несовместимого анализа имеет три разновидности:

- ✓ неповторяемое чтение;
- ✓ проблема строк-призраков (фантомы);
- ✓ собственно несовместимый анализ.



# ПРОБЛЕМЫ ПАРАЛЛЕЛИЗМА ТРАНЗАКЦИЙ

*Неповторяемое чтение* возникает тогда, когда между двумя, подряд идущими, операциями чтения транзакции 1 вклинивается транзакция 2, изменяющее считываемое значение в строке таблицы.

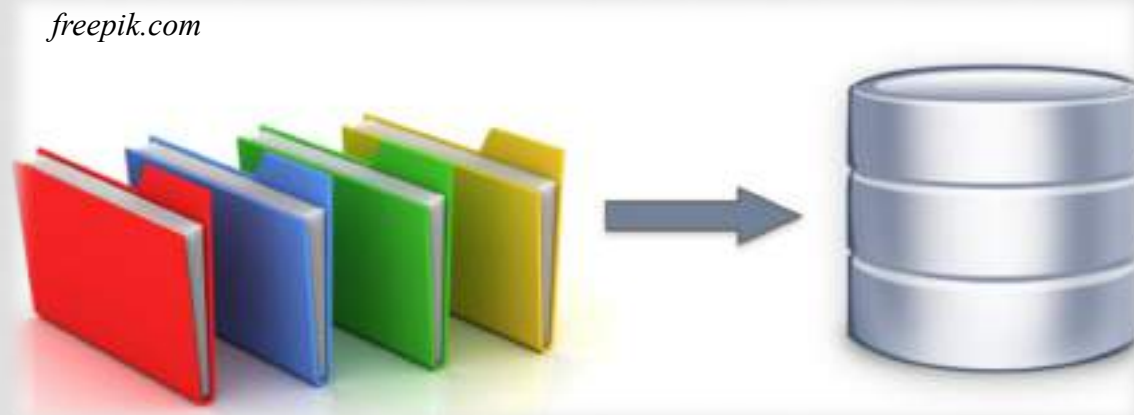


# ПРОБЛЕМЫ ПАРАЛЛЕЛИЗМА ТРАНЗАКЦИЙ

Транзакция 1		Транзакция 2
SELECT f2 ( <i>f2=40</i> ) FROM tbl1 WHERE f1=1;	<i>t1</i>	
	<i>t2</i>	SELECT f2 ( <i>f2=40</i> ) FROM tbl1 WHERE f1=1;
UPDATE tbl1 SET f2=F2-30 ( <i>f2=10</i> ) WHERE f1=1;	<i>t3</i>	
COMMIT	<i>t4</i>	
	<i>t5</i>	SELECT f2 ( <i>f2=10</i> ) FROM tbl1 WHERE f1=1;
	<i>t6</i>	COMMIT

# ПРОБЛЕМЫ ПАРАЛЛЕЛИЗМА ТРАНЗАКЦИЙ

*Фантомы* появляются в результате вставки новой строки (строк) транзакцией 1 по условию  $U1$  между двумя, подряд идущими, выборками, удовлетворяющими условию  $U1$  и выполняемыми транзакцией 2.



# ПРОБЛЕМЫ ПАРАЛЛЕЛИЗМА ТРАНЗАКЦИЙ

Транзакция 1		Транзакция 2
	<i>t1</i>	SELECT SUM(f2) FROM tbl1 Where f1=10.05.14;
INSERT INTO tbl1 (f1,f2) VALUES (10.05.14,20);	<i>t2</i>	
COMMIT	<i>t3</i>	
	<i>t4</i>	SELECT f2 FROM tbl1 Where f1=10.01.14;
	<i>t5</i>	COMMIT

# ПРОБЛЕМЫ ПАРАЛЛЕЛИЗМА ТРАНЗАКЦИЙ

Транзакция 1		Транзакция 2
SELECT Cч1 ( <i>Cч1=40</i> ) FROM tbl1 WHERE f1=1;	t1	
SUM=SUM+ Cч1 ( <i>SUM=40</i> )	t2	
SELECT Cч2 ( <i>Cч2=50</i> ) FROM tbl1 WHERE f1=1;	t3	
SUM=SUM+ Cч2 ( <i>SUM=90</i> )	t4	
	t5	SELECT Cч3 ( <i>Cч3=30</i> ) FROM tbl1 WHERE f1=1;
	t6	UPDATE tbl1 SET Cч3= Cч3-10 ( <i>Cч3=20</i> );



# ПРОБЛЕМЫ ПАРАЛЛЕЛИЗМА ТРАНЗАКЦИЙ

Транзакция 1		Транзакция 2
	<i>t7</i>	SELECT Cч1 ( <i>Cч1=40</i> ) FROM tbl1 WHERE f1=1;
	<i>t8</i>	UPDATE tbl1 SET Cч1= Cч1+10 ( <i>Cч1=50</i> );
	<i>t9</i>	COMMIT
SELECT Cч3 ( <i>Cч3=20</i> ) FROM tbl1 WHERE f1=1;	<i>t10</i>	
SUM=SUM+ Cч3 ( <i>SUM=110</i> )	<i>t11</i>	
COMMIT		

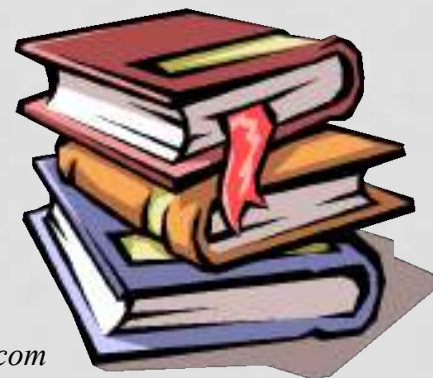
## ЗАДАНИЯ ДЛЯ СРС

1. Для чего предназначен механизм транзакций при работе с БД? Приведите примеры.
2. Перечислите свойства транзакций.
3. С какой целью проводится журнализация изменений БД? Объясните схему журнализации.
4. В чем заключается проблема параллелизма транзакций?
5. Приведите классификацию проблем параллелизма транзакций.

# РЕКОМЕНДУЕМАЯ ЛИТЕРАТУРА

1. Т. Конноли, К. Бегг. Базы данных: Проектирование, реализация и сопровождение. Теория и практика [Текст] : учебное пособие / 3-е изд. - М. ; СПб. ; Киев : Вильямс, 2018. - 1440 с.

2. К.Дж. Дейт. Введение в системы баз данных. М., 2018. - 1328 с.



***ЛЕКЦИЯ ОКОНЧЕНА.  
БЛАГОДАРЮ ЗА ВНИМАНИЕ!***