

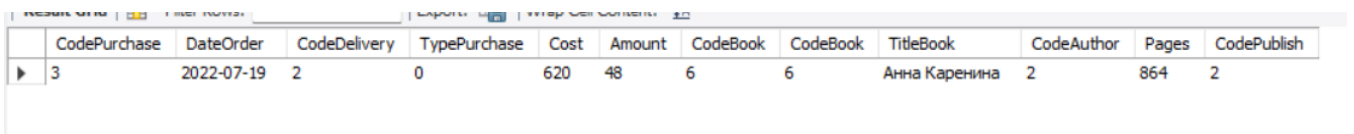
Отчет

Хранимые процедуры

MySQL

1. Вывести все сведения о поставке (все поля таблицы Purchases), а также название книги (поле Title_book) с максимальной общей стоимостью (использовать поля Cost и Amount)

```
CREATE DEFINER='root'@'localhost' PROCEDURE `max_cost`()
begin
select * from purchases p
join books b on b.CodeBook = p.CodeBook
where p.Cost*p.Amount = (select max(Amount*Cost) from purchases);
end
call max_cost();
```



	CodePurchase	DateOrder	CodeDelivery	TypePurchase	Cost	Amount	CodeBook	CodeBook	TitleBook	CodeAuthor	Pages	CodePublish
▶	3	2022-07-19	2	0	620	48	6	6	Анна Каренина	2	864	2

Выбираем максимальную общую стоимость в подзапросе, присоединяем к таблице purchases таблицу books и выбираем те записи, в которых стоимость равна максимальной стоимости.

2. Сосчитать количество книг определенного автора (ФИО автора является входным параметром).

```
CREATE DEFINER='root'@'localhost' PROCEDURE `authors_books_count`(n
Char(30))
begin
select count(p.CodeBook) from books p
join authors a on a.CodeAuthor = p.CodeAuthor
where a.NameAuthor = n;
end
call authors_books_count('Пушкин А.С.');
```

Result Grid		Filter Rows:
	count(p.CodeBook)	
▶	4	

Присоединяем таблицу authors к таблице books, подсчитываем количество книг у которых имя автора равно входному значению n.

3. Определить адрес определенного поставщика (Наименование поставщика является входным параметром, адрес поставщика – выходным параметром)

```
CREATE DEFINER='root'@'localhost' PROCEDURE `get_delivers_address`(in n
Char(30),out address varchar(100))
```

```
begin
```

```
select Address into address from delivers d
```

```
where d.NameDelivery = n;
```

```
end
```

```
set @ deliveryAddress = ";
```

```
call get_delivers_address('Иванов И.И.',@deliveryAddress);
```

```
select @ deliveryAddress;
```

	@deliveryAddress
▶	пр. Октябрьский, 456

Выбираем из поставщиков тех, у кого имя совпадает с введенным параметром, записываем полученный адрес в выходной параметр

4. Выполните операцию вставки в таблицу Books. Код книги должен увеличиваться автоматически на единицу.

```
CREATE DEFINER='root'@'localhost' PROCEDURE `insert_new_book`(in title
varchar(45), in codeauthor int, in pages int, in cPubl int )
```

```
begin
```

```
insert into books
```

```
values ((select max(v.CodeBook) from books v)+1 ,title,codeauthor, pages, cPubl);
end
call insert_new_book('new book',1,234,3);
```

36 09:00:20 call insert_new_book('new book',1,234,3) 1 row(s) affected

Result Grid | Filter Rows: | Edit: | Export/Import:

	CodeBook	TitleBook	CodeAuthor	Pages	CodePublish
15	15	Хамелеон	7	12	2
16		Хамелеон	7	12	2
17		Хамелеон	7	12	2
18		new book	1	234	3
	NULL	NULL	NULL	NULL	NULL

Добавляем новую запись в таблицу книг, взяв входные параметры, считаем максимальный код книги в таблице и увеличиваем на единицу.

- Определить поставки с минимальной и максимальной стоимостью книг. Отобразить список всех поставок. Если стоимость поставки – максимальная, то вывести сообщение «Максимальная стоимость», если стоимость – минимальная, то вывести сообщение «Минимальная стоимость», иначе – «Средняя стоимость».

```
CREATE DEFINER='root'@'localhost' PROCEDURE `min_max_purchase`()
begin
  SELECT
    *,
    CASE
      WHEN (Amount * Cost) = (SELECT MAX(Amount * Cost) FROM
purchases) THEN 'Максимальная стоимость'
      WHEN (Amount * Cost) = (SELECT MIN(Amount * Cost) FROM
purchases) THEN 'Минимальная стоимость'
      ELSE 'Средняя стоимость'
    END AS CostMessage
  FROM
    purchases;
end
call min_max_purchase();
```

	CodePurchase	DateOrder	CodeDelivery	TypePurchase	Cost	Amount	CodeBook	CostMessage
▶	1	2022-08-12	1	0	240	20	4	Средняя стоимость
	2	2024-03-03	1	1	560	1	2	Средняя стоимость
	3	2022-07-19	2	0	620	48	6	Максимальная стоимость
	4	2020-03-27	1	1	137	3	8	Средняя стоимость
	5	2024-05-15	1	1	240	2	4	Средняя стоимость
	6	2021-05-23	2	1	315	1	9	Средняя стоимость
	7	2024-05-15	2	0	450	30	3	Средняя стоимость
	8	2021-11-13	2	1	190	2	10	Средняя стоимость
	9	2022-09-22	3	0	560	23	2	Средняя стоимость
	10	2023-04-14	3	1	176	1	1	Средняя стоимость
	11	2024-02-22	3	1	390	1	5	Средняя стоимость
	12	2024-11-15	3	1	164.4	1	8	Средняя стоимость
	13	2019-05-05	3	0	120	9	13	Средняя стоимость
	42	2024-12-13	2	1	2	4	5	Минимальная стоимость

Считаем максимальную стоимость книги в purchases и минимальную стоимость, стоимость каждой книги сравниваем с максимальной и минимальной и выводим соответствующее сообщение

6. Определить количество записей в таблице поставщиков. Пока записей меньше 10, делать в цикле добавление записи в таблицу с автоматическим наращиванием значения ключевого поля, а вместо названия поставщика ставить значение 'не известен'.

```
CREATE DEFINER='root'@'localhost' PROCEDURE `insert_delivers_up_10`
(
)
begin
set @rows = (select count(CodeDelivery) from delivers);
while @rows <=10 do
insert into delivers(CodeDelivery, NameDelivery)
values((select max(d.CodeDelivery) from delivers d)+1, "не известен");
set @rows = @rows +1;
end while;
end
call insert_delivers_up_10();
```

	CodeDelivery	NameDelivery	NameCompany	Address	Phone	INN
	1	Иванов И.И.	ОАО "Литера"	пр.Октябрьский, 456	89161234567	7701123456
	2	Смирнова С.А.	ЗАО "ЛитроДом"	ул. Советская, д. 22, оф. 10, Екатеринбург	89514325678	6645765432
	3	Кузнецов К.К.	АО Читательский Клуб	пр. Мира, д. 56, корп. 2, Санкт-Петербург	89217654321	7802234567
	4	Сидорова П.Д.	ОАО "БукХаус"	нет сведений	81231231231	7865434322
▶	5	не известен	NULL	NULL	NULL	
	6	не известен	NULL	NULL	NULL	NULL
	7	не известен	NULL	NULL	NULL	NULL
	8	не известен	NULL	NULL	NULL	NULL
	9	не известен	NULL	NULL	NULL	NULL
*	NULL	NULL	NULL	NULL	NULL	NULL

В переменную записать изначальное количество строк, пока их меньше 10 вставляем новые записи и прибавляем к переменной единицу

PostgreSQL

1. Вывести фамилии и имена студентов (поля Surname, Name из таблицы Students) с максимальным средним баллом за весь период обучения (условие по полю Estimate из таблицы Progress).

```
begin
return query
select surname,stud_name,lastname, avg(estimate)
from students
join groups_stud using (code_group)
join progress using (code_stud)
group by surname,stud_name,lastname, name_group
having avg(estimate) =
(select max(Avg_es) from (select(avg(estimate)) as Avg_es from progress
group by code_stud));
end
select * from max_avg_estimate();
```

	surname_ character	name_ character	lnnm character	avg_ numeric
1	Михайлов	Андрей	Андреевич	5.0000000000000000

Подсчитываем средний балл для каждого студента, среди них выбираем наибольший, сравниваем средний балл каждого студента с максимальным средним

2. Определить средний балл определенного студента (ФИО студента является входным параметром).

```
begin
return (select avg(estimate) from progress
join students using (code_stud)
where surname = surname_ and stud_name = name_ and lastname = lastname_);
```

end

```
select * from avg_my_stud('Рябова','Екатерина','Сергеевна');
```

	avg_my_stud numeric
1	4.0000000000000000

Выбираем средний балл студента, у которого совпадает ФИО с входными параметрами. Вызываем процедуру.

3. Определить специальность и номер курса определенного студента (ФИО студента является входным параметром, Название специальности и Номер курса – выходными параметрами).

begin

```
select name_speciality, num_group into special, course from groups_stud
```

```
join students using (code_group)
```

```
where surname = surname_ and stud_name = name_ and lastname = lastname_;
```

end

```
select * from course_spec_stud('Рябова','Екатерина','Сергеевна');
```

	special character	course integer
1	Информационные технологии	2

Выбираем название специальности, номер курса, записываем в переменную у студента с необходимым ФИО

4. Выполните операцию вставки в таблицу Students. Код студента должен автоматически увеличиваться на единицу.

begin

```
insert into students
```

```
values ((select count(f.code_stud) from students f)+1,surname_,name_,  
lastname_,code_group_,birthday_,phone_);
```

end

call insert_stud('Студенов','Степан','Станиславович',101,'2007-07-07',8343546354);

1	101A	Бадамшина	...	Аделина	Ришатовна	101	2006-03-27	23435353	4.5000000000000000
2	101C	Беляева		Ксения	Ришатовна	101	2006-05-18	346546542	3.0000000000000000
3	101T	Михайлов		Андрей	Андреевич	101	2006-07-14	32423423	5.0000000000000000
4	102S	Рябова		Екатерина	...	102	2008-02-21	4567654	4.0000000000000000
5	104K	Соколов		Дмитрий	Иванович	104	2008-12-12	2343234	4.0000000000000000
6	104U	Уранова		Роза	нет сведений	104	2005-07-23	72345456	[null]
7	106N	Кудрявцев	...	Виктор	Викторович	106	2007-01-04	74564232	4.0000000000000000
8	107R	Лебедев		Евгений	Евгеньевич	107	2005-10-09	3453443	3.5000000000000000
9	108E	Иванов		Павел	Сергеевич	[null]	[null]	[null]	[null]
10	109K	student		student	[null]	[null]	[null]	[null]	3.0000000000000000
11	12	Студенов		Степан	Станиславович	101	2007-07-07	8343546354	0

Вставляем в таблицу студентов новую запись с данными из входных параметров, считаем количество студентов и прибавляем единицу для увеличения кода.

5. Определить средний возраст всех студентов. Вывести список всех студентов. Если возраст студента больше среднего возраста, то вывести сообщение «Вы старше среднего возраста всех студентов», если возраст – меньше, то вывести сообщение «Ваш возраст меньше среднего возраста всех студентов», а иначе – «Ваш возраст равен среднему возрасту всех студентов».

begin

return query

select distinct surname, stud_name,extract(Year from age(current_date, birthday))
age,

case when (extract(Year from age(current_date, birthday))) < (
select avg(extract(Year from age(current_date,
birthday)))
from students)

then 'Ваш возраст меньше среднего возраста всех студентов'
when (extract(Year from age(current_date, birthday))) > (
select avg(extract(Year from age(current_date,
birthday)))
from students)

then 'Вы старше среднего возраста всех студентов'

```

else 'Ваш возраст равен среднему возрасту всех студентов'
end as Messag
from students;
end
select * from avg_age_stud();

```

	surname_ character		name_ character		age numeric		message_ text
1	student		student		[null]		Ваш возраст равен среднему возрасту всех студентов
2	Бадамшина ...		Аделина ...		19		Вы старше среднего возраста всех студентов
3	Беляева		Ксения ...		19		Вы старше среднего возраста всех студентов
4	Иванов		Павел ...		[null]		Ваш возраст равен среднему возрасту всех студентов
5	Кудрявцев ...		Виктор ...		18		Ваш возраст равен среднему возрасту всех студентов
6	Лебедев		Евгений ...		19		Вы старше среднего возраста всех студентов
7	Михайлов ...		Андрей ...		18		Ваш возраст равен среднему возрасту всех студентов
8	Рябова		Екатерина ...		17		Ваш возраст меньше среднего возраста всех студентов
9	Соколов		Дмитрий ...		16		Ваш возраст меньше среднего возраста всех студентов
10	Студенов		Степан ...		17		Ваш возраст меньше среднего возраста всех студентов
11	Уранова		Роза		19		Вы старше среднего возраста всех студентов

Считаем максимальный и минимальный средний возраст, сравниваем средний возраст каждого студента и выводим соответствующее сообщение.

- Определить количество записей в таблице дисциплин. Пока записей меньше 10, делать в цикле добавление записи в таблицу с автоматическим наращиванием значения ключевого поля, а вместо названия дисциплины ставить значение 'не известно'.

```

DECLARE

```

```

    current_count INTEGER;

```

```

BEGIN

```

```

    SELECT COUNT(code_subject) INTO current_count FROM subject;

```

```

    WHILE current_count < 10 LOOP

```

```

        INSERT INTO subject (code_subject, name_subject)

```

```

        VALUES ((SELECT COALESCE(MAX(s.code_subject), 0) + 1 FROM subject
s), 'не известно');

```

```

        current_count := current_count + 1;

```


END LOOP;

END;

call insert_subject_to_10();

	code_subject [PK] integer	name_subject character (20)	count_hours integer
1	21002	Электротехника...	110
2	21003	Экономика	116
3	21004	Статистика	80
4	21005	Иностранный я...	58
5	21001	Мат анализ	64
6	21006	не известно	[null]
7	21007	не известно	[null]
8	21008	не известно	[null]
9	21009	не известно	[null]
10	21010	не известно	[null]

В переменную записываем изначальное количество строк, пока их меньше десяти вставляем новые записи с предметами и увеличиваем счетчик строк на единицу.

Триггеры

MySQL

1. Создайте триггер, запускаемый при занесении новой строки в таблицу Авторы. Триггер должен увеличивать счетчик числа добавленных строк.

```
CREATE DEFINER='root'@'localhost' TRIGGER `authors_BEFORE_INSERT`  
BEFORE INSERT ON `authors` FOR EACH ROW BEGIN
```

```
    set new.CodeAuthor = (select max(a.CodeAuthor) +1 from authors a);
```

```
END
```

insert into authors

values (0,"Фет А.А.",'1999-09-09',1);

59	Фет А.А.	1999-09-09	1
NULL	NULL	NULL	NULL

Создаем триггер перед вставкой новой записи, считаем максимальный код автора, увеличиваем на единицу и записываем как новое значение.

2. Добавьте в таблицу Авторы поле Количество книг (Count_books) целого типа со значением по умолчанию 0. Создайте хранимую процедуру, которая подсчитывает количество книг по каждому автору и заносит в поле Count_books эту информацию. Создайте триггер, запускаемый после внесения новой информации о книге.

```
CREATE DEFINER='root'@'localhost' PROCEDURE `count_authors_books`()  
  
begin  
  
update authors a  
  
set a.CountBooks = (select count(TitleBook) from books  
  
join (select * from authors) b on b.CodeAuthor = books.CodeAuthor  
  
where a.CodeAuthor = b.CodeAuthor  
  
group by b.CodeAuthor) ;  
  
end
```

```
CREATE DEFINER='root'@'localhost' TRIGGER `books_AFTER_INSERT` AFTER  
INSERT ON `books` FOR EACH ROW BEGIN  
  
call count_authors_books;  
  
END
```

6	Твен М.	1835-11-30	NULL
7	Чехов А.П.	1860-01-29	6
8	Шолохов М.А.	1905-05-24	1

```
insert into books  
  
values ((select max(h.CodeBook) from books h)+1,'Хамелеон',7,12,2);
```

6	Твен М.	1835-11-30	NULL
7	Чехов А.П.	1860-01-29	7
8	Шолохов М.А.	1905-05-24	1

Создаем процедуру, где считаем количество книг каждого автора. Создаем триггер после вставки где вызывается процедур для подсчета.

3. Создайте триггер, запускаемый при внесении информации о новых поставках. Выполните проверку о количестве добавляемой книги в таблице

Книги. Если количество экземпляров книг в таблице меньше 10, то необходимо увеличить стоимость книг на 20 %.

```
CREATE DEFINER='root'@'localhost' TRIGGER `purchases_BEFORE_INSERT`  
BEFORE INSERT ON `purchases` FOR EACH ROW BEGIN
```

```
if NEW.amount < 10 then
```

```
    set NEW.Cost = NEW.Cost * 1.2;
```

```
end if;
```

```
END
```

```
insert into purchases
```

```
values (13,'2019-05-05',3,0,100,9,13);
```

8	2021-11-13	2	1	190	2	10
9	2022-09-22	3	0	560	23	2
10	2023-04-14	3	1	176	1	1
11	2024-02-22	3	1	390	1	5
12	2024-11-15	3	1	164.4	1	8
13	2019-05-05	3	0	120	9	13

Создание триггера до вставки , если количество книг в заказе меньше 10, то стоимость заказа увеличивается на 20%

4. Запретить вставлять новые строки в таблицу Поставщики, выводя при этом сообщение «Вставка строк запрещена»

```
CREATE DEFINER='root'@'localhost' TRIGGER `delivers_BEFORE_INSERT`  
BEFORE INSERT ON `delivers` FOR EACH ROW BEGIN
```

```
signal sqlstate '45000'
```

```
set message_text = 'Вставка строк запрещена';
```

```
END
```

```
insert into delivers
```

```
values(11,'Анатолийев А.Б.', 'БыстроДост','ул. Маркова  
д.34',12345545,'32384932697');
```

```
insert into delivers values(11,'Анатолийев А.Б.', 'БыстроДост','ул. Маркова д.34',12345545,'323849326... Error Code: 1644. Вставка строк запрещена
```

Триггер до вставки вызываем ошибку с сообщением

PostgreSQL

1. Создайте триггер, запускаемый при занесении новой строки в таблицу Преподаватели. Триггер должен увеличивать счетчик числа добавленных строк

begin

```
new.code_lector = (select max(l.code_lector)+1 from lectors l);
```

```
return new;
```

end

insert into lectors

values (0,'Андреев Андрей Андреевич','к.т.н.','преподаватель','2015-12-12');

6	6	Петров Сергей Петрович	д.ф.м.н.	декан	2002-02-27
7	7	Власова Мария Степановна	д.т.н.	заведующий кафе...	2006-01-04
8	8	Петров Савелий Яковлевич	к.т.н.	[null]	[null]
9	9	Алексеев Алексей Алексеевич	...	преподаватель	2015-05-05
10	10	Дымова Юлия Викторовна	д.м.н	преподаватель	2019-05-15
11	11	Андреев Андрей Андреевич	к.т.н.	преподаватель	2015-12-12

Создаем триггер перед вставкой новой записи, считаем максимальный код преподавателя, увеличиваем на единицу и записываем как новое значение.

2. Добавьте в таблицу Студенты поле Средний балл (Avg_Estimate) вещественного типа со значением по умолчанию 0. Создайте хранимую процедуру, которая подсчитывает средний балл для каждого студента и заносит в поле Avg_Estimate эту информацию. Создайте триггер, запускаемый после внесения новой информации об оценках студента и автоматически обновляет информацию о среднем балле студента.

begin

update students

```
set avg_estimate = (select avg(estimate) from progress
```

```
group by code_stud
```

```
having students.code_stud = progress.code_stud);
```

end

begin

```

call avg_estimate_for_students();

return new;

end

```

```

call avg_estimate_for_students();

```

```

select * from students;

```

```

insert into progress
values (12,'107R',4,21003,'2024-12-27',3);

```

	code_stud [PK] character (10)	surname character (20)	stud_name character (20)	lastname character (20)	code_group integer	birthday date	phone numeric (11)	avg_estimate numeric
1	101A	Бадамшина ...	Аделина	Ришатовна	101	2006-03-27	23435353	4.5000000000000000
2	101C	Беляева	Ксения	Ришатовна	101	2006-05-18	346546542	3.0000000000000000
3	101T	Михайлов	Андрей	Андреевич	101	2006-07-14	32423423	5.0000000000000000
4	102S	Рябова	Екатерина ...	Сергеевна	102	2008-02-21	4567654	4.0000000000000000
5	104K	Соколов	Дмитрий	Иванович	104	2008-12-12	2343234	4.0000000000000000
6	104U	Уранова	Роза	нет сведений	104	2005-07-23	72345456	[null]
7	106N	Кудрявцев ...	Виктор	Викторович	106	2007-01-04	74564232	4.0000000000000000
8	107R	Лебедев	Евгений	Евгеньевич	107	2005-10-09	3453443	3.5000000000000000

Создаем процедуру для подсчета среднего балла каждого студента и записи в таблицу, создаем триггер где вызывается процедура

3. Создайте триггер, запускаемый при внесении информации о новых оценках. Выполните проверку наличия информации о добавляемом студенте в таблице Студенты. Если данная информация в таблице отсутствует, то необходимо запустить хранимую процедуру на вставку записи в таблицу Студенты (параметры можно задать произвольно).

```

begin

if not (select exists(

        select 1 from students

        where code_stud = new.code_stud ))

then call insert_student_(new.code_stud,'student','student');

end if;

```

```
return new;
```

```
end
```

```
select * from progress;
```

```
insert into progress
```

```
values(13,'109K',1,21004,'2024-12-21',3);
```

11	109K	student	student	[null]	[null]	[null]	[null]	3.0000000000000000
----	------	---------	---------	--------	--------	--------	--------	--------------------

Создаем процедуру, выполняем проверку если кода из таблицы оценок нет в таблице студентов, то добавляем новую запись с произвольными данными.

4. Запретить вставлять новые строки в таблицу Группы, выводя при этом сообщение «Вставка строк запрещена».

```
begin
```

```
raise exception 'Вставка строк запрещена';
```

```
return new;
```

```
end
```

```
insert into groups_stud
```

```
values(1,'c',3,'fjnfj');
```

Data Output Messages Notifications

ERROR: Вставка строк запрещена

CONTEXT: PL/pgSQL function decline_inserting() line 2 at RAISE

SQL state: P0001

Триггер до вставки , вызываем сообщение об ошибке при попытке вставить новую информацию

Транзакции

MySQL

1. Проверьте выполнение команд транзакции при добавлении новой информации об издательствах

```
start transaction;
```

```
select * from publishinghouse;
```

```
insert into publishinghouse
```

```
values((select coalesce(max(p.CodePublish), 0) + 1 from publishinghouse as p),  
'ООО "Эстмо"', 'Казань');
```

```
savepoint publishHouse_save_point;
```

```
insert into publishinghouse
```

```
values(0, 'ООО "Абвгд"', 'Казань');
```

```
rollback to savepoint publishHouse_save_point;
```

```
commit;
```

```
select * from publishinghouse;
```

	CodePublish	Publish	City
▶	1	АСТ	Москва
	2	Азбука	Санкт-Петербург
	3	Мартин	Тверь
	4	Эксмо	Москва
	5	Питерсофт	Санкт-Петербург
	13	МИФ	Москва
	14	NULL	Москва
	15	ООО "Эстмо"	Казань
*	NULL	NULL	NULL

В транзакции вставляем новую запись в таблицу издательств, ставим точку сохранения, добавляем еще новую запись, откатываемся к точке сохранения

PostgreSQL

1. Проверьте выполнение команд транзакции при добавлении новой информации о преподавателях.

```
start transaction;
```

```
select * from lectors;
```

```
insert into lectors
```

```
values((select coalesce(max(code_lector), 0) + 1 from lectors), 'Сапова Алла Викторовна', 'д.м.н', 'преподаватель', '2019-05-15');
```

```
savepoint lectors_save_point;
```

```
insert into lectors
```

```
values(0, 'Сапова Анна Семеновна', 'д.м.н', 'преподаватель', '2019-05-15');
```

```
rollback to savepoint lectors_save_point;
```

```
commit;
```

```
end transaction;
```

6	6	Петров Сергей Петрович	...	д.ф.м.н.	декан	2002-02-27
7	7	Власова Мария Степановна	...	д.т.н.	заведующий кафе...	2006-01-04
8	8	Петров Савелий Яковлевич	...	к.т.н.	[null]	[null]
9	9	Алексеев Алексей Алексеевич	...	к.ф.н.	преподаватель	2015-05-05
10	10	Дымова Юлия Викторовна	...	д.м.н	преподаватель	2019-05-15
11	11	Андреев Андрей Андреевич	...	к.т.н.	преподаватель	2015-12-12
12	12	Дымова Юлия Викторовна	...	д.м.н	преподаватель	2019-05-15
13	13	Сапова Алла Викторовна	...	д.м.н	преподаватель	2019-05-15

В транзакции вставляем новую запись в таблицу преподавателей, ставим точку сохранения, добавляем еще новую запись, откатываемся к точке сохранения

Работа с пользователем

1. Администратор – обладает всеми правами

```
create user admin_db@localhost identified by 'admin1234';
```

```
grant all privileges on *.* to admin_db@localhost with grant option;
```

```
flush privileges;
```

```
show grants for admin_db@localhost;
```

	Grants for admin_db@localhost
►	GRANT SELECT, INSERT, UPDATE, DELETE, CR...
	GRANT APPLICATION_PASSWORD_ADMIN,AU...

Задаем права на все таблицы всех бд на любые действия для админа

2. Диспетчер – просматривает, заполняет и изменяет справочники: книги, авторы, издательства, поставщики.

```
grant select, insert, update on library_db.authors to dispatcher_db@localhost;
```

```
grant select, insert, update on library_db.books to dispatcher_db@localhost;
```

```
grant select, insert, update on library_db.delivers to dispatcher_db@localhost;
```

```
grant select, insert, update on library_db.publishinghouse to dispatcher_db@localhost;
```

```
flush privileges;
```

Grants for dispatcher_db@localhost	
▶	GRANT USAGE ON *.* TO 'dispatcher_db'@'localhost'
	GRANT SELECT, INSERT, UPDATE ON `library_db`.`authors` TO 'dispatcher_db'@'localhost'
	GRANT SELECT, INSERT, UPDATE ON `library_db`.`books` TO 'dispatcher_db'@'localhost'
	GRANT SELECT, INSERT, UPDATE ON `library_db`.`delivers` TO 'dispatcher_db'@'localhost'
	GRANT SELECT, INSERT, UPDATE ON `library_db`.`publishinghouse` TO 'dispatcher_db'@'localhost'

Задаем права на просмотр, вставку и редактирование таблиц книги, авторы, поставщики и издательства для бд библиотека

3. Менеджер по работе с поставщиками – просматривает и добавляет новую информацию в справочники, оформляет поставки.

```
grant select, insert, update on library_db.authors to dispatcher_db@localhost;
```

```
grant select, insert, update on library_db.books to dispatcher_db@localhost;
```

```
grant select, insert, update on library_db.delivers to dispatcher_db@localhost;
```

```
grant select, insert, update on library_db.publishinghouse to dispatcher_db@localhost;
```

```
flush privileges;
```

Grants for manager_delivers@localhost	
	GRANT USAGE ON *.* TO 'manager_delivers'@'localhost'
	GRANT SELECT, INSERT ON `library_db`.`authors` TO 'manager_delivers'@'localhost'
	GRANT SELECT, INSERT ON `library_db`.`books` TO 'manager_delivers'@'localhost'
	GRANT SELECT, INSERT ON `library_db`.`delivers` TO 'manager_delivers'@'localhost'
	GRANT SELECT, INSERT ON `library_db`.`publishinghouse` TO 'manager_delivers'@'localhost'
	GRANT SELECT, INSERT, UPDATE ON `library_db`.`purchases` TO 'manager_delivers'@'localhost'

Менеджеру задаем права на просмотр и добавление для таблиц авторы, книги, поставщики и издательства для бд библиотека

4. Поставщики – просматривают только свои поставки

```
create user supplier@localhost identified by 'supplier1234';  
grant select on library_db.show_my_purchases to supplier@localhost;  
flush privileges;  
show grants for supplier@localhost;
```

	Grants for supplier@localhost
►	GRANT USAGE ON *.* TO `supplier` @ `localhost`
	GRANT SELECT ON `library_db`.`show_my_purchases` TO `supplier` @ `localhost`

Задаем права на просмотр своих поставок