Хранимые процедуры

MySql:

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).

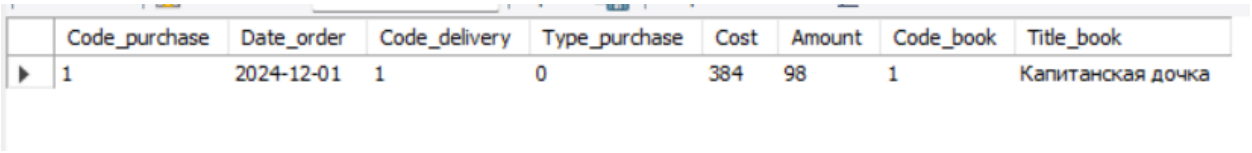
```
delimiter //
create procedure GetMaxCost()
begin
   select purchases.*, books.Title_book from purchases
   join books on purchases.Code_book = books.Code_book
   where purchases.Cost * purchases.Amount = (select max(Cost * Amount) from purchases);
end //
```

Соединяем таблицы Books и Purchases по Code_book, вычисляем стоимость каждой покупки, находим макс общую стоимость.

```
call books.GetMaxCost();
```

Вызываем хранимую процедуру

Результат:



Code_purchase	Date_order	Code_delivery	Type_purchase	Cost	Amount	Code_book	Title_book
1	2024-12-01	1	0	384	98	1	Капитанская дочка

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

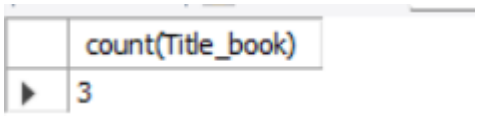
```
delimiter //
create procedure CountBook(in NameAuthor varchar(50))
begin
        select count(Title_book) from books join authors on
   books.Code_author = authors.Code_author
   where authors.Name_author = NameAuthor;
end //
```

Задаем входной параметр имя автора, соединяем таблицы Authors и Books и находим кол-во книг конкретного автора из бд.

call books.CountBook('Пушкин А.С.');

Вызываем хранимую процедуру

Результат:



	count(Title_book)
▶	3

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

delimiter //

create procedure AddressDelivery(in NameDeliv varchar(40), out AddressDeliv varchar(60))

begin

select deliveries.Address into AddressDeliv from deliveries

where deliveries.Name_delivery = NameDeliv;

end //

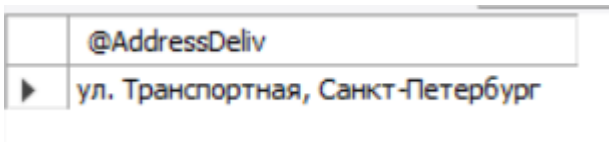
Задаем входной параметр Название поставщика, выходной Адрес поставщика, находим конкретного поставщика из бд.

set @AddressDeliv = '';

call books.AddressDelivery('Скорость', @AddressDeliv);

select @AddressDeliv;

Вызываем



	@AddressDeliv
▶	ул. Транспортная, Санкт-Петербург

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

delimiter //

create procedure InsertIntoTBook(in TitleB varchar(40), in CodeA int, in PagesB int, in CodePublish int)

begin

declare max_code int;

select coalesce(max(Code_book), 0) + 1 into max_code from books;

insert into books(Code_book, Title_book, Code_author, Pages, Code_publish)

values(max_code,TitleB, CodeA, PagesB, CodePublish);

end //

Задаем переменную max_code для хранения максимального кода книги, увеличиваем код книги на одну от максимальной. Вставляем новую книгу

call books.InsertIntoTBook(Золотая рыбка, 1, 23, 3);

Вызываем процедуру

Code_book	Title_book	Code_author	Pages	Code_publish
2	Преступление и н...	2	331	3
3	Темные аллеи	3	2	4
4	Война и мир	4	1270	2
5	Отцы и дети	5	300	4
6	Мертвые души	6	352	1
7	Мастер и Маргарита	7	416	1
8	Вишневый сад	8	66	2
9	Гранатовый браслет	9	48	3
10	Король, дама, валет	10	288	4
11	Наука. Техника. И...	4	230	2
12	Руслан и Людмила	1	352	3
13	Сказка о царе Сал...	1	30	1
14	Золотая рыбка	1	23	3
NULL	NULL	NULL	NULL	NULL

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».

delimiter //

create procedure MinMaxPurchases()

begin

declare max_cost double;

declare min_cost double;

```
        select max(Cost * Amount), min(Cost * Amount) into max_cost, min_cost
        from purchases;
        select purchases.*,
        case
                when (Cost * Amount) = max_cost then "Максимальная стоимость"
        when (Cost * Amount) = min_cost then "Минимальная стоимость"
        else "Средняя стоимость"
         end as Costs from purchases;
    end //
```

Объявляем переменные мин и макс стоимость, принимаем макс и мин стоимость из бд умножая цену на кол-во, затем выводим сообщения для самой дорогой поставки дешевой и для остальных поставок.

```
call books.MinMaxPurchases();
```

Вызываем процедуру

Code_purchase	Date_order	Code_delivery	Type_purchase	Cost	Amount	Code_book	Costs
1	2024-12-01	1	0	384	98	1	Максимальная стоимость
2	2023-10-13	2	0	120	43	3	Средняя стоимость
3	2023-11-19	3	1	411	1	6	Минимальная стоимость
4	2024-03-27	1	1	520	22	2	Средняя стоимость
5	2024-11-13	2	1	276	21	8	Средняя стоимость
6	2023-06-01	4	1	543	1	9	Средняя стоимость
7	2024-08-07	3	1	120	34	3	Средняя стоимость
8	2024-12-05	4	0	400	20	4	Средняя стоимость
9	2023-01-02	1	1	960	9	1	Средняя стоимость

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'.

```
delimiter //
create procedure CountDeliveries()
begin
        declare max_code int;
   declare delivery_count int;
```

```
select coalesce(max(Code_delivery), 0) + 1 into max_code from deliveries;
select count(*) into delivery_count from deliveries;
    while delivery_count < 10 do
        insert into deliveries(Code_delivery, Name_delivery)
        values(max_code, 'не известен');
        set max_code = max_code + 1;
        set delivery_count = delivery_count + 1;
  end while;
end //
```

Находим макс код поставщика, проходим по циклу пока код меньше 10 то вставляем данные и увеличиваем макс код и кол-во

```
call books.CountDeliveries();
```

Вызываем процедуру

Code_delivery	Name_delivery	Name_company	Address	Phone	INN
1	Быстрее всех	ООО "Экспресс"	ул. Логистическая, Москва	74951234567	7701234567
2	Скорость	ЗАО "Транссервис"	ул. Транспортная, Санкт-Петербург	78129876543	7804567890
3	Устойчивый путь	ИП Иванов Иван	нет сведений	78431234567	1234567890
4	На время	ОАО "Логистик Групп"	пр.Наукоград,Новосибирск	73834567890	5401234567
5	не известен	NULL	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
10	не известен	NULL	NULL	NULL	NULL
NULL	NULL	NULL	NULL	NULL	NULL

PostgreSql:

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 s.Surname, s.Namee
  FROM Students s
  JOIN Progress p ON s.Code_stud = p.Code_stud
  GROUP BY s.Code_stud, s.Surname, s.Namee
```

```
    HAVING AVG(p.Estimate) = (
        SELECT MAX(AvgEstimate)
        FROM (
            SELECT AVG(p.Estimate) as AvgEstimate
            FROM Progress p
            GROUP BY p.Code_stud
        ) AS MaxAvg
    );
end;
```

Присоединяем таблицу прогресс для оценки, вычисляем средние оценки (AVG(p.Estimate)) для каждого студента, затем находим макс значение из этих средних (MAX(AvgEstimate)), выбираем записи, где средняя оценка студента равна этому максимальному значению

```
select * from MaxScore();
```

Вызываем процедуру

	surname character	namee character
1	Васильев	Сергей
2	Михайлов	Дмитрий
3	Иванов	Павел
4	Сидоров	Алексей

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

```
declare avg_estimate numeric(3,2);
begin
    select avg(estimate) into avg_estimate from students s
    join progress p on s.Code_stud = p.Code_stud
    where s.Surname = "Surname" and s.Namee = "Namee" and s.Lastname = "Lastname";
    return avg_estimate;
end;
```

Находим среднюю оценку каждого студента, и выводим среднюю оценку конкретного студента из бд

select * from public."Avg_estimate"('Бадамшина', 'Аделина', 'Ришатовна');

Вызываем процедуру

	Avg_estimate numeric
1	4.50

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

begin

select Name_course, Name_speciality into "Name_course", "Name_speciality" from groupss g

join students s on g.Code_group = s.Code_group

where s.Surname = "Surname" and s.Namee= "Namee" and s.Lastname = "Lastname";

end;

Выводим название спец и номер курса конкретного студента

select * from public."Info_Student"('Бадамшина', 'Аделина', 'Ришатовна');

Вызываем процедуру

	Name_course character	Name_speciality character
1	3	Программист ...

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

begin

INSERT INTO Students (

Code_stud, Surname, Namee, Lastname, Code_group, Birthday, Phone

)

VALUES

((select cast(max(cast(Code_stud as INTEGER)) + 1 as VARCHAR) from Students),

"Surname", "Namee", "Lastname", "Code_group", "Birthday", "Phone");

end;

Объявляем входные параметры, вычисляем макс код студента и прибавляем 1 и вставляем другие параметры

call public."Insert_Students"('Петрова', 'Яна', 'Андреевна', 0, '2006-09-13', '79870987654');

Вызываем процедуру

2	1	Петрова	...	Яна	Андреевна	...	1	2006-09-13	79111234566	3.0000000000000000
3	11	Козлов	...	Владимир	...	Дмитриевич	...	0	2006-02-28	79201234577	4.0000000000000000
4	12	Орлова	...	Екатерина	...	Алексеевна	...	7	2007-10-11	79211234578	[default]
5	13	Борисов	...	Александр	...	Сергеевич	...	4	2004-07-07	79221234579	[default]
6	14	Иванов	...	Павел	...	Сергеевич	...	[null]	[null]	[null]	5.0000000000000000
7	15	Иванов	...	Павел	...	Сергеевич	...	[null]	[null]	[null]	[default]
8	16	Бадамшина	...	Аделина	...	Ришатовна	...	0	2006-03-28	79870987654	[default]
9	18	Иванов	...	Павел	...	Сергеевич	...	1	2006-09-13	79111234566	3.0000000000000000
10	19	Петрова	...	Яна	Андреевна	...	0	2006-09-13	79870987654	0

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 s.Code_stud, s.Surname, s.Namee, s.Lastname,

extract(year from age(now(), Birthday)) as Age,

case

WHEN extract(year from age(now(), Birthday)) > avg_age then 'Вы старше среднего возраста всех студентов'

WHEN extract(year from age(now(), Birthday)) < avg_age then 'Ва возраст меньше среднего возраста всех студентов'

else 'Ваш возраст равен среднему возрасту всех студентов'

end as age_message

from students s,

(select avg(extract(year from age(now(), Birthday))) as avg_age from students) as avg_table;

end;

Вычисляем из даты рождения кол-во лет, находим средний возраст по всем студентам и выводим сообщения для студентов старше среднего возраста младше и возраст равный среднему

select * from minmaxage();

Вызываем процедуру

	code_stud character	surname character	namee character	lastname character	age numeric	age_message text
1	0	Бадамши...	Аделина	Ришатов...	19	Вы старше среднего возраста всех студентов
2	1	Петрова	Яна	Андреевн...	18	Ва возраст меньше среднего возраста всех студентов
3	11	Козлов	Владими...	Дмитрие...	19	Вы старше среднего возраста всех студентов
4	12	Орлова	Екатерин...	Алексеев...	17	Ва возраст меньше среднего возраста всех студентов
5	13	Борисов	Александ...	Сергееви...	20	Вы старше среднего возраста всех студентов
6	2	Сидоров	Алексей	Петрович...	20	Вы старше среднего возраста всех студентов
7	3	Сидоров	Алексей	Петрович...	18	Ва возраст меньше среднего возраста всех студентов
8	4	Кузнецов...	Елена	Владими...	17	Ва возраст меньше среднего возраста всех студентов
9	5	Михайло...	Дмитрий	Алексеев...	19	Вы старше среднего возраста всех студентов
10	6	Николаев...	Ольга	Ивановна...	18	Ва возраст меньше среднего возраста всех студентов
11	7	Васильев...	Сергей	Николаев...	15	Ва возраст меньше среднего возраста всех студентов
12	8	Смирнов...	Татьяна	Викторов...	17	Ва возраст меньше среднего возраста всех студентов
13	14	Иванов	Павел	Сергееви...	[null]	Ваш возраст равен среднему возрасту всех студент...
14	9	Федоров	Андрей	Михайло...	18	Ва возраст меньше среднего возраста всех студентов
15	15	Иванов	Павел	Сергееви...	[null]	Ваш возраст равен среднему возрасту всех студент...
16	16	Бадамши...	Аделина	Ришатов...	19	Вы старше среднего возраста всех студентов
17	18	Иванов	Павел	Сергееви...	18	Ва возраст меньше среднего возраста всех студентов
18	19	Петрова	Яна	Андреевн...	18	Ва возраст меньше среднего возраста всех студентов

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 max_code integer;

subject_count integer;

begin

```
select coalesce(max(Code_subject), 0) + 1 into max_code from subjects;
select count(*) into subject_count from subjects;
while subject_count < 15 loop
        insert into subjects(Code_subject, Name_subject, Count_hours)
        values(max_code, 'не известен', 0);
        max_code = max_code + 1;
        subject_count = subject_count + 1;
    end loop;
end;
```

Находим макс код предмета проходимся по циклу до количества книг равный 15 и добавляем новые записи в таблицу предметы увеличивая код и кол-во

```
call count_s();
```

Вызываем процедуру

	code_subject [PK] integer	name_subject character (25)	count_hours integer
1	0	Обществознание	56
2	1	Иностранный язык	48
3	2	Архитектура	72
4	3	Веб-разработка	128
5	4	Физ-ра	86
6	5	Математический анализ	212
7	6	Дискрет лог	78
8	7	Операционные системы	120
9	8	Экономика	64
10	9	Программирование	256
11	10	не известен	0
12	11	не известен	0
13	12	не известен	0
14	13	не известен	0
15	14	не известен	0

Триггеры

MySql:

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.Code_author = (select max(Code_author) + 1 from authors);

END

Увеличиваем макс код на 1 до вставки записи в таблицу Авторов

insert into authors(name_author, birthday)

values('Пастернак','1900-01-01');

Вставляем новую запись

Code_author	Name_author	Birthday	Count_books
1	Пушкин А.С.	1799-06-06 00:00:00	4
2	Достоевский Ф.М.	1821-10-30 00:00:00	1
3	Бунин И.А.	1870-10-10 00:00:00	1
4	Толстой Л.Н.	1828-08-28 00:00:00	3
5	Тургенев И.С.	1818-10-28 00:00:00	1
6	Гоголь Н.В.	1809-03-20 00:00:00	1
7	Булгаков М.А.	1891-05-03 00:00:00	1
8	Чехов А.П.	1860-01-17 00:00:00	1
9	Куприн А.И.	1870-08-26 00:00:00	1
10	Набоков В.В.	1899-04-10 00:00:00	1
11	Чехов	1789-05-07 00:00:00	0
12	Куприн	1879-03-27 00:00:00	0
13	Куприн	1879-02-27 00:00:00	0
14	Пастернак	1900-01-01 00:00:00	0
NULL	NULL	NULL	NULL

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 `CounterBook`()

begin

update authors

set count_books = (select count(Title_book) from books

where books.Code_author = authors.Code_author);

end

Подсчитываем кол-во книг конкретного автора и обновляем столбец count_books

insert into books

values(15, 'Спящая царевна', 1, 12, 2);

Вставляем новую запись в таблицу books

До:

Code_author	Name_author	Birthday	Count_books
1	Пушкин А.С.	1799-06-06 00:00:00	4
2	Достоевский Ф.М.	1821-10-30 00:00:00	1
3	Бунин И.А.	1870-10-10 00:00:00	1
4	Толстой Л.Н.	1828-08-28 00:00:00	3
5	Тургенев И.С.	1818-10-28 00:00:00	1
6	Гоголь Н.В.	1809-03-20 00:00:00	1
7	Булгаков М.А.	1891-05-03 00:00:00	1
8	Чехов А.П.	1860-01-17 00:00:00	1
9	Куприн А.И.	1870-08-26 00:00:00	1
10	Набоков В.В.	1899-04-10 00:00:00	1
11	Чехов	1789-05-07 00:00:00	0
12	Куприн	1879-03-27 00:00:00	0
13	Куприн	1879-02-27 00:00:00	0
14	Пастернак	1900-01-01 00:00:00	0
NULL	NULL	NULL	NULL

После:

Code_author	Name_author	Birthday	Count_books
1	Пушкин А.С.	1799-06-06 00:00:00	5
2	Достоевский Ф.М.	1821-10-30 00:00:00	1
3	Бунин И.А.	1870-10-10 00:00:00	1
4	Толстой Л.Н.	1828-08-28 00:00:00	3
5	Тургенев И.С.	1818-10-28 00:00:00	1
6	Гоголь Н.В.	1809-03-20 00:00:00	1
7	Булгаков М.А.	1891-05-03 00:00:00	1
8	Чехов А.П.	1860-01-17 00:00:00	1
9	Куприн А.И.	1870-08-26 00:00:00	1
10	Набоков В.В.	1899-04-10 00:00:00	1
11	Чехов	1789-05-07 00:00:00	0
12	Куприн	1879-03-27 00:00:00	0
13	Куприн	1879-02-27 00:00:00	0
14	Пастернак	1900-01-01 00:00:00	0
NULL	NULL	NULL	NULL

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 + (new.cost * 0.2);

end if;

END

Увеличиваем стоимость книги на 20% если кол-во меньше 10

INSERT INTO purchases

VALUES (9, '2023-01-02', 1, 1, 800, 9, 1);

Вставляем новую запись в таблицу поставок

Code_purchase	Date_order	Code_delivery	Type_purchase	Cost	Amount	Code_book
1	2024-12-01	1	0	384	98	1
2	2023-10-13	2	0	120	43	3
3	2023-11-19	3	1	411	1	6
4	2024-03-27	1	1	520	22	2
5	2024-11-13	2	1	276	21	8
6	2023-06-01	4	1	543	1	9
7	2024-08-07	3	1	120	34	3
8	2024-12-05	4	0	400	20	4
9	2023-01-02	1	1	960	9	1
NULL	NULL	NULL	NULL	NULL	NULL	NULL

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

CREATE DEFINER=`root`@`localhost` TRIGGER `deliveries_BEFORE_INSERT` BEFORE INSERT ON `deliveries` FOR EACH ROW BEGIN

SIGNAL SQLSTATE '45000'

SET MESSAGE_TEXT = 'Вставка строк запрещена';

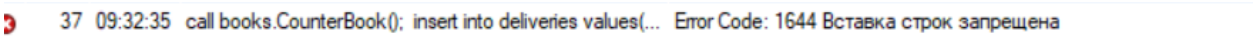
END

Выводим сообщение о запрете вставки новых записей в таблицу поставщиков

insert into deliveries

values(11, 'Быстрый', 'ЗАО Экспресс', 'ул. Достоевского 15', 79870440560, 12345678900);

Пытаемся добавить новую запись



Проверьте выполнение команд транзакции при добавлении новой информации об издательствах.

start transaction;

select * from publishing_house;

savepoint publishing_house_4;

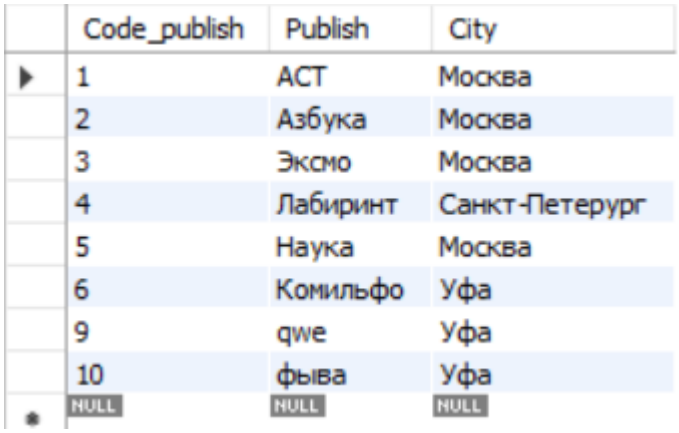
insert into publishing_house (Publish, City)

values('ячсмитьбьти', 'Уфа');

rollback to savepoint publishing_house_4;

Запускаем транзакцию находим издательство под кодом 4 сохраняем изменения, затем добавляем новую запись и откатываем изменения до точки сохранения.

Code_publish	Publish	City
1	АСТ	Москва
2	Азбука	Москва
3	Эксмо	Москва
4	Лабиринт	Санкт-Петерург
5	Наука	Москва
6	Комильфо	Уфа
9	qwe	Уфа
10	фыва	Уфа
11	ячсмить...	Уфа
NULL	NULL	NULL



Code_publish	Publish	City
1	АСТ	Москва
2	Азбука	Москва
3	Эксмо	Москва
4	Лабиринт	Санкт-Петерург
5	Наука	Москва
6	Комильфо	Уфа
9	qwe	Уфа
10	фыва	Уфа
NULL	NULL	NULL

PostgreSql:

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

```
begin
    new.Code_lector = (select max(Code_lector) + 1 from lectors);
    return new;
end;
```

Увеличиваем код преподавателя на 1 от максимального кода

```
insert into lectors(Name_lector, Science, Post, Date)
values('Иванов Сергей Иванович', 'Кандидат наук', 'преподаватель', '2016-10-13');
```

Вставляем новую запись в таблицу lectors

	code_lector [PK] integer	name_lector character (40)	science character (25)	post character (25)	date date
1	0	Иванов Иван Иванович	Кандидат нау...	преподавател...	2015-10-13
2	1	Петрова Анна Сергеевна	Кандидат нау...	профессор	2010-01-12
3	2	Ахметова Айгуль Мутагаровна	Кандидат нау...	преподавател...	2017-09-01
4	3	Михайлов Дмитрий Алексееви...	Кандидат нау...	декан	2009-11-28
5	4	Николаева Ольга Ивановна	Доктор наук	декан	2018-09-01
6	5	Смирнова Татьяна Викторовна...	Кандидат нау...	старший преп...	2011-11-11
7	6	Федоров Андрей Михайлович	Доктор наук	преподавател...	2020-12-20
8	7	Николаева Ольга Ивановна	Кандидат нау...	преподавател...	2017-03-27
9	8	Петров Савелий Яковлевич	к.т.н.	[null]	[null]
10	9	Петров Савелий Яковлевич	к.т.н.	[null]	[null]
11	10	Иванов Сергей Иванович	Кандидат нау...	преподавател...	2016-10-13

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
                        where students.Code_stud = progress.Code_stud);
end;
```

Хранимая процедура для расчёта среднего балла

```
begin
call avg_est();
return new;
end;
```

Вызываем хранимую процедуру в триггере

```
insert into progress
values('18', 7, 4, '2023-06-20', 5, 17);
```

добавляем новую запись

	code_stud [PK] character (25)	surname character (25)	namee character (25)	lastname character (25)	code_group integer	birthday date	phone numeric	avg_estimate numeric
1	0	Бадамшина	Аделина	Ришатовна	0	2006-03-27	79870440560	4.5000000000000000
2	1	Петрова	Яна	Андреевна	1	2006-09-13	79111234566	3.0000000000000000
3	11	Козлов	Владимир	Дмитриевич	0	2006-02-28	79201234577	4.0000000000000000
4	12	Орлова	Екатерина	Алексеевна	7	2007-10-11	79211234578	[default]
5	13	Борисов	Александр	Сергеевич	4	2004-07-07	79221234579	[default]
6	14	Иванов	Павел	Сергеевич	[null]	[null]	[null]	5.0000000000000000
7	15	Иванов	Павел	Сергеевич	[null]	[null]	[null]	[default]
8	16	Бадамшина	Аделина	Ришатовна	0	2006-03-28	79870987654	[default]
9	18	Иванов	Павел	Сергеевич	1	2006-09-13	79111234566	4.0000000000000000
10	19	Петрова	Яна	Андреевна	0	2006-09-13	79870987654	[default]
11	2	Сидоров	Алексей	Петрович	2	2005-01-14	79121234569	4.0000000000000000
12	3	Сидоров	Алексей	Петрович	2	2006-11-30	79121234569	5.0000000000000000
13	4	Кузнецова	Елена	Владимировн...	3	2008-03-12	79131234570	4.0000000000000000
14	5	Михайлов	Дмитрий	Алексеевич	4	2005-09-18	79141234571	5.0000000000000000
15	6	Николаева	Ольга	Ивановна	5	2007-01-25	79151234572	3.0000000000000000
16	7	Васильев	Сергей	Николаевич	6	2009-08-10	79161234573	5.0000000000000000
17	8	Смирнова	Татьяна	Викторовна	7	2007-06-05	79171234574	3.0000000000000000
18	9	Федоров	Андрей	Михайлович	8	2006-12-20	79181234575	4.0000000000000000

15	18	7	4	2023-06-20	3	16
16	18	7	4	2023-06-20	5	17

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
    call insert_st(new.Code_stud);
    return new;
end;
```

Вызываем хранимую процедуру в триггере

```
begin
    if not exists(select 1 from students where Code_stud = code_stud1) then
    insert into students
    values(Code_stud1, 'Иванов', 'Павел', 'Сергеевич', 1, '2006-09-13', 79111234566,0);
    end if;
```

end;

добавляем нового студента только если его ещё нет в базе

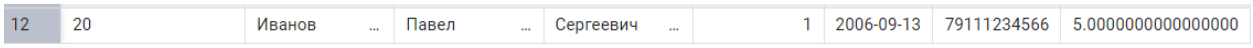
insert into progress

values('20', 7, 4, '2023-06-20', 5, 19);

Вставили новую запись с несуществующим студентом

18	20	7	4	2023-06-20	5	19

Добавился новый студент по умолчанию



12	20	Иванов ...	Павел ...	Сергеевич ...	1	2006-09-13	79111234566	5.0000000000000000

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

begin

raise exception 'Вставка строк запрещена' using errcode = '45000';

end;

insert into groupss

values(10,'22П-1',3,'Программист');

Пытаемся вставить новую строку

```
ERROR:  Вставка строк запрещена
CONTEXT:  PL/pgSQL function exception_f() line 2 at RAISE
```

Проверьте выполнение команд транзакции при добавлении новой информации о преподавателях.

start transaction;

select * from lectors;

savepoint lectors2;

insert into lectors

values((select coalesce(max(Code_lector), 0) + 1 from lectors), 'Бадамшина Аделина Ришатовна', 'Доктор наук', 'преподаватель', '2022-05-01');

rollback to savepoint lectors2;

запускаем транзакцию делаем выборку всех препод. Сохраняем изменения вставляем новую запись, откатываем изменения до точки сохранения

После вставки:

	code_lector [PK] integer	name_lector character (40)	science character (25)	post character (25)	date date
1	0	Иванов Иван Иванович	Кандидат нау...	преподавател...	2015-10-13
2	1	Петрова Анна Сергеевна	Кандидат нау...	профессор	2010-01-12
3	4	Николаева Ольга Ивановна	Доктор наук	декан	2018-09-01
4	5	Смирнова Татьяна Викторовн...	Кандидат нау...	старший преп...	2011-11-11
5	6	Федоров Андрей Михайлович...	Доктор наук	преподавател...	2020-12-20
6	7	Николаева Ольга Ивановна	Кандидат нау...	преподавател...	2017-03-27
7	8	Петров Савелий Яковлевич	к.т.н.	[null]	[null]
8	9	Петров Савелий Яковлевич	к.т.н.	[null]	[null]
9	10	Иванов Сергей Иванович	Кандидат нау...	преподавател...	2016-10-13
10	2	Ахметова Айгуль Мутагаровн...	Кандидат нау...	преподавател...	2017-09-01
11	3	Михайлов Дмитрий Алексеев...	Кандидат нау...	декан	2009-11-28
12	11	Бадамшина Аделина Ришато...	Доктор наук	преподавател...	2022-05-01
13	12	Юмагужина Диана Ильдаров...	Доктор наук	преподавател...	2022-05-01

После отката:

	code_lector [PK] integer	name_lector character (40)	science character (25)	post character (25)	date date
1	0	Иванов Иван Иванович	Кандидат нау...	преподавател...	2015-10-13
2	1	Петрова Анна Сергеевна	Кандидат нау...	профессор	2010-01-12
3	4	Николаева Ольга Ивановна	Доктор наук	декан	2018-09-01
4	5	Смирнова Татьяна Викторовна...	Кандидат нау...	старший преп...	2011-11-11
5	6	Федоров Андрей Михайлович	Доктор наук	преподавател...	2020-12-20
6	7	Николаева Ольга Ивановна	Кандидат нау...	преподавател...	2017-03-27
7	8	Петров Савелий Яковлевич	к.т.н.	[null]	[null]
8	9	Петров Савелий Яковлевич	к.т.н.	[null]	[null]
9	10	Иванов Сергей Иванович	Кандидат нау...	преподавател...	2016-10-13
10	2	Ахметова Айгуль Мутагаровна	Кандидат нау...	преподавател...	2017-09-01
11	3	Михайлов Дмитрий Алексееви...	Кандидат нау...	декан	2009-11-28
12	11	Бадамшина Аделина Ришатов...	Доктор наук	преподавател...	2022-05-01

Работа с пользователями

Администратор – обладает всеми правами

CREATE USER 'dbadmin'@'localhost' IDENTIFIED BY '12345';

GRANT ALL PRIVILEGES ON books.* TO 'dbadmin'@'localhost';

SHOW GRANTS FOR 'dbadmin'@'localhost';

```
GRANT USAGE ON *.* TO `dbadmin`@`localhost`
GRANT ALL PRIVILEGES ON `books`.* TO `dbadmin`@`localhost`
```

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

CREATE USER 'dbdispetcher'@'localhost' IDENTIFIED BY 'qwe123';

GRANT INSERT, SELECT, UPDATE ON books.authors TO 'dbdispetcher'@'localhost';

GRANT INSERT, SELECT, UPDATE ON books.books TO 'dbdispetcher'@'localhost';

GRANT INSERT, SELECT, UPDATE ON books.publishing_house TO 'dbdispetcher'@'localhost';

GRANT INSERT, SELECT, UPDATE ON books.deliveries TO 'dbdispetcher'@'localhost';

SHOW GRANTS FOR 'dbdispetcher'@'localhost';

```
GRANT USAGE ON *.* TO `dbdispetcher`@`localhost`
GRANT SELECT, INSERT, UPDATE ON `books`.`authors` TO `dbdispetcher`@`localhost`
GRANT SELECT, INSERT, UPDATE ON `books`.`books` TO `dbdispetcher`@`localhost`
GRANT SELECT, INSERT, UPDATE ON `books`.`deliveries` TO `dbdispetcher`@`localhost`
GRANT SELECT, INSERT, UPDATE ON `books`.`publishing_house` TO `dbdispetcher`@`localhost`
```

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

CREATE USER 'dbmanager'@'localhost' IDENTIFIED BY '0987';

GRANT INSERT, SELECT ON books.authors TO 'dbmanager'@'localhost';

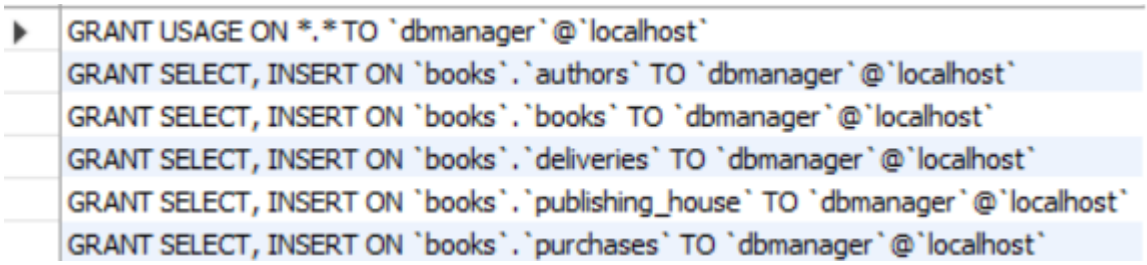
GRANT INSERT, SELECT ON books.books TO 'dbmanager'@'localhost';

GRANT INSERT, SELECT ON books.publishing_house TO 'dbmanager'@'localhost';

GRANT INSERT, SELECT ON books.deliveries TO 'dbmanager'@'localhost';

GRANT INSERT, SELECT ON books.purchases TO 'dbmanager'@'localhost';

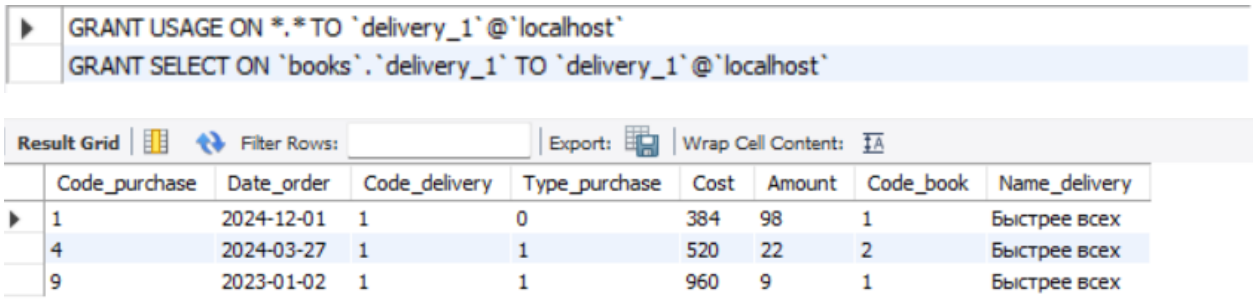
SHOW GRANTS FOR 'dbmanager'@'localhost';



Поставщики – просматривают только свои поставки

```
CREATE VIEW delivery_1 AS
SELECT p.*, d.Name_delivery
FROM Purchases p
JOIN Deliveries d ON p.Code_delivery = d.Code_delivery
WHERE p.Code_delivery = 1;

CREATE VIEW delivery_2 AS
SELECT p.*, d.Name_delivery
FROM Purchases p
JOIN Deliveries d ON p.Code_delivery = d.Code_delivery
WHERE p.Code_delivery = 2;

CREATE VIEW delivery_3 AS
SELECT p.*, d.Name_delivery
FROM Purchases p
JOIN Deliveries d ON p.Code_delivery = d.Code_delivery
WHERE p.Code_delivery = 3;

CREATE VIEW delivery_4 AS
SELECT p.*, d.Name_delivery
FROM Purchases p
```

```
JOIN Deliveries d ON p.Code_delivery = d.Code_delivery
WHERE p.Code_delivery = 4;

CREATE USER 'delivery_1'@'localhost' IDENTIFIED BY 'qwe';
CREATE USER 'delivery_2'@'localhost' IDENTIFIED BY 'asd';
CREATE USER 'delivery_3'@'localhost' IDENTIFIED BY 'zxc';
CREATE USER 'delivery_4'@'localhost' IDENTIFIED BY 'ewq';

GRANT SELECT ON books.delivery_1 TO 'delivery_1'@'localhost';
GRANT SELECT ON books.delivery_2 TO 'delivery_2'@'localhost';
GRANT SELECT ON books.delivery_3 TO 'delivery_3'@'localhost';
GRANT SELECT ON books.delivery_4 TO 'delivery_4'@'localhost';

SHOW GRANTS FOR 'delivery_1'@'localhost';
```



```
GRANT USAGE ON *.* TO `delivery_1`@`localhost`
GRANT SELECT ON `books`.`delivery_1` TO `delivery_1`@`localhost`
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

Code_purchase	Date_order	Code_delivery	Type_purchase	Cost	Amount	Code_book	Name_delivery
1	2024-12-01	1	0	384	98	1	Быстрее всех
4	2024-03-27	1	1	520	22	2	Быстрее всех
9	2023-01-02	1	1	960	9	1	Быстрее всех

Программа(объед датасет)

Main_habr:

```python
import ...

app = FastAPI()

with open('model_lr_habr5.pkl', 'rb') as file:
    model = pickle.load(file)

with open('vectorizer_habr5.pkl', 'rb') as file:
    vectorizer = pickle.load(file)

1 usage
def fun_punctuation_text(text):
    text = text.lower()
    text = ''.join([ch for ch in text if ch not in string.punctuation])
    text = ''.join([i if not i.isdigit() else '' for i in text])
    text = ''.join([i if i.isalpha() else ' ' for i in text])
    text = re.sub( pattern: r'\s+', repl: ' ', text, flags=re.I)
    text = re.sub( pattern: '[a-z]', repl: '', text, flags=re.I)
    st = '\xa0'
    text = ''.join([ch if ch not in st else ' ' for ch in text])
    return text
```

```python
1 usage
def fun_lemmatizing_text(text):
    tokens = word_tokenize(text)
    res = list()
    for word in tokens:
        p = pymorphy3.MorphAnalyzer(lang='ru').parse(word)[0]
        res.append(p.normal_form)
    text = " ".join(res)
    return text

1 usage
def fun_tokenize(text):
    nltk.download('stopwords')
    russian_stopwords = nltk.corpus.stopwords.words("russian")
    russian_stopwords.extend(
        ['т.д.', 'т', 'д', 'это','который','с','своём','всем', 'свой', 'всё', 'весь', '”', '“', '…', 'привет', 'хабр', 'сегодня'
    t = word_tokenize(text)
    tokens = [token for token in t if token not in russian_stopwords]
    text = " ".join(tokens)
    return text

1 usage
def fun_pred_text(text):
    text = fun_punctuation_text(text)
    text = fun_lemmatizing_text(text)
    text = fun_tokenize(text)
    return text
```

```python
def predict_cluster(text):
    text_vectorized = vectorizer.transform([fun_pred_text(text)])
    probabilities = model.predict_proba(text_vectorized)[0]

    mapping = {
        0: 'Цифровые сервисы и регулирование рынка: от агрегаторов такси до образовательных платформ',
        1: 'Технологии будущего и их влияние на человека: от робототехники до нейронаук',
        2: 'Современные подходы к работе, обучению и IT-разработке',
        3: 'Цифровой дизайн, технологии и брендинг в современном бизнесе',
        4: 'IT-образование и профессиональная разработка программного обеспечения'
    }

    probabilities_list = []
    for cluster_num, prob in enumerate(probabilities):
        cluster_name = mapping[cluster_num]
        probabilities_list.append({
            "cluster_name": cluster_name,
            "probability": float(prob)
        })

    return probabilities_list

1 usage
class Item(BaseModel):
    text: str

@app.post("/predict")
def post_predict_text(item: Item):
    return {'probabilities': predict_cluster(item.text)}
```

App_streamlit:

```python
import streamlit as st
import requests
import os

os.environ['HTTP_PROXY'] = ''
os.environ['HTTPS_PROXY'] = ''

st.title("Предсказание кластера статьи")
input_text = st.text_area("Введите описание статьи", height=200)

if st.button("Предсказать"):
    if input_text == "":
        st.write("Введите текст")
    else:
        data = {
            "text": input_text
        }
        url = 'http://127.0.0.1:8000/predict'
        response = requests.post(url, json=data)
        result = response.json()
        probabilities = result.get("probabilities", [])

        st.markdown("""
        ### Вероятности по кластерам
        """)

        for prob in probabilities:
            st.write(f"{prob['cluster_name']}: {prob['probability']}")
```

Предсказание кластера статьи

Введите описание статьи

Шаблоны в C++ опасны для начинающего пользователя не столько сложностью, сколько опосредованными, неочевидными последствиями для проекта в целом.

Среди проблем есть и замедление компиляции, и увеличение размера объектных файлов, и искажение структуры проекта из-за невозможности разделить шаблонные классы на объявления и определения также легко, как мы делаем это с обычными классами.

Далее мы рассмотрим несколько способов решить перечисленные сложности за счет более

Предсказать

Вероятности по кластерам

Цифровые сервисы и регулирование рынка: от агрегаторов такси до образовательных платформ: 0.009333046688514 76

Технологии будущего и их влияние на человека: от робототехники до нейронаук: 0.05393961252041171

Современные подходы к работе, обучению и IT-разработке: 0.033602321149653704

Цифровой дизайн, технологии и брендинг в современном бизнесе: 0.01565196775717549

IT-образование и профессиональная разработка программного обеспечения: 0.8874730518842444

Предсказание кластера статьи

Введите описание статьи

Технические собеседования, без сомнения, являются важнейшим этапом взаимодействия работодателей с соискателями. Специалист может проявлять свои коммуникативные навыки в высшей степени удачно, но если речь идёт о технической должности, песок в глаза бросить не удастся.

В современном найме в сфере информационных технологий сегодня существует проблема определения квалификации соискателя. Диплом не может служить пасскодом, а опыт работы в прошлых компаниях - лишь отчасти. Решением "наверняка" представляется лишь

Предсказать

Вероятности по кластерам

Цифровые сервисы и регулирование рынка: от агрегаторов такси до образовательных платформ: 0.015356222469486905

Технологии будущего и их влияние на человека: от робототехники до нейронаук: 0.21195903168268485

Современные подходы к работе, обучению и IT-разработке: 0.6685253365579146

Цифровой дизайн, технологии и брендинг в современном бизнесе: 0.026406526347931628

IT-образование и профессиональная разработка программного обеспечения: 0.07775288294198211

Предсказание кластера статьи

Введите описание статьи

Доброго времени суток, «Хабр»!

Сегодня я стану вашим гидом по выбору лучших нейросетей-синонимайзеров 2025 года. Давайте узнаем, какие инструменты помогут преобразить ваш текст.

Синонимайзер — это инструмент, который позволяет перефразировать текст, подбирая синонимы и изменяя его структуру. Он применяется для SEO-копирайтинга, написания

Предсказать

Вероятности по кластерам

Цифровые сервисы и регулирование рынка: от агрегаторов такси до образовательных платформ: 0.022506703869194233

Технологии будущего и их влияние на человека: от робототехники до нейронаук: 0.5379343004376262

Современные подходы к работе, обучению и IT-разработке: 0.21704659370255136

Цифровой дизайн, технологии и брендинг в современном бизнесе: 0.03935603287429613

IT-образование и профессиональная разработка программного обеспечения: 0.18315636911633196