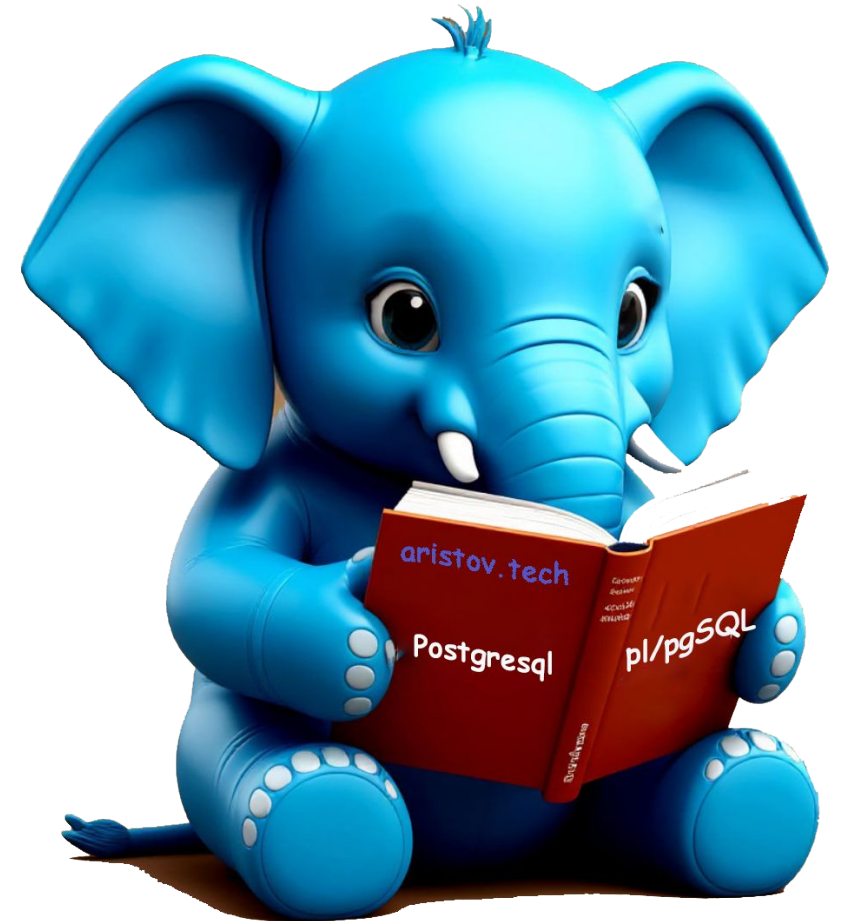


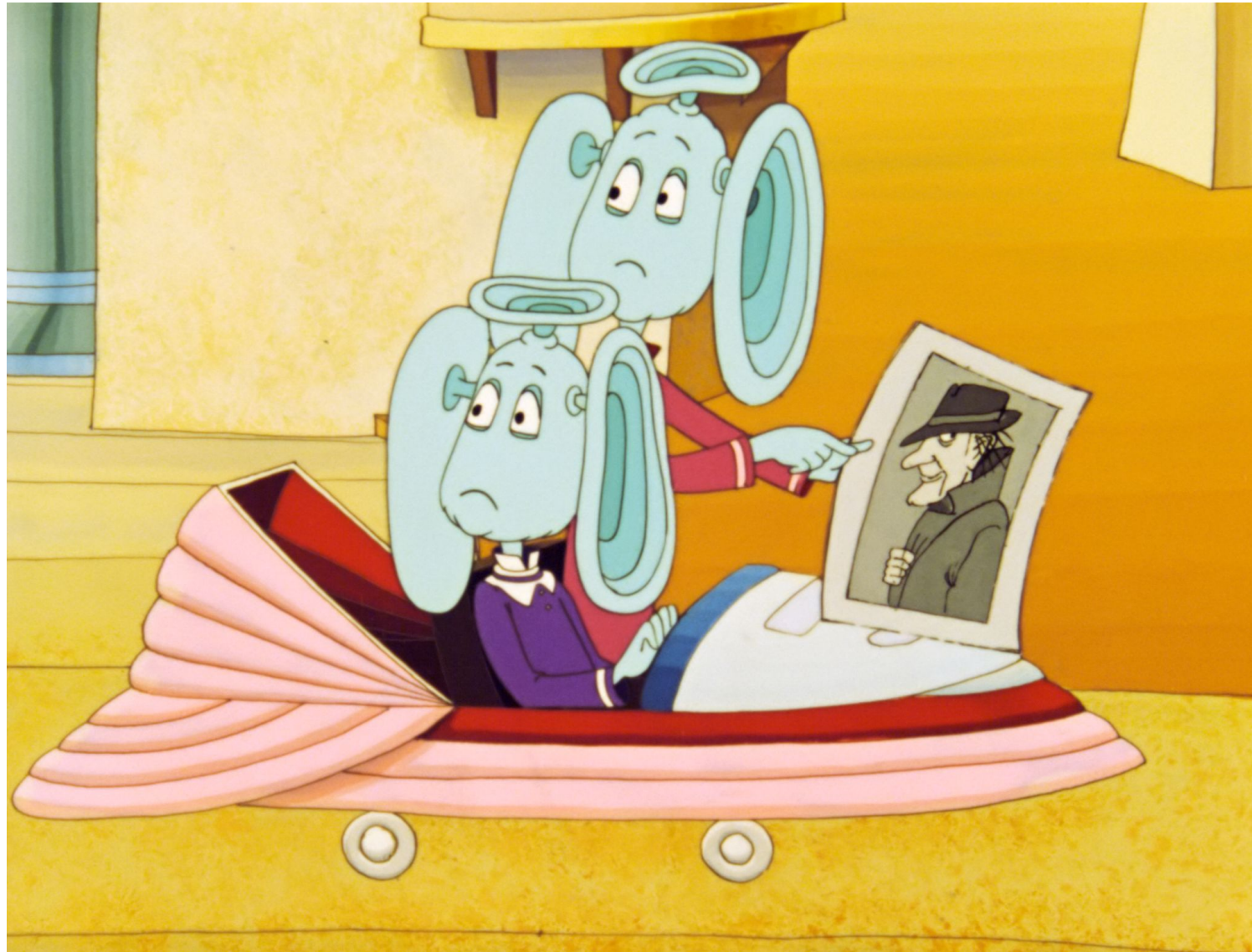
Аристов Евгений

PL/pgSQL в PostgreSQL

за 31 занятие

Обработка ошибок





**Аристов
Евгений
Николаевич**



<https://aristov.tech>

<https://aristov.tech>

Founder & CEO aristov.tech

25 лет занимаюсь разработкой БД и ПО

Архитектор высоконагруженных баз данных и инфраструктуры

Спроектировал и разработал более ста проектов для финансового сектора, сетевых магазинов, фитнес-центров, отелей.

Сейчас решаю актуальные для бизнеса задачи: аудит и оптимизация БД и инфраструктуры, миграция на PostgreSQL, обучение сотрудников.

Автор более 10 практических курсов по PostgreSQL, MySQL, Mongo и др..

Автор книг по PostgreSQL. Новинка [PostgreSQL 16: лучшие практики оптимизации](#)

Правила вебинара

Задаем вопрос в чат

Вопросы вижу, отвечу в момент логической паузы

Если есть вопрос голосом - поставьте знак ? в чат

Если остались вопросы, можно их задать на следующем занятии или в комментариях к записи

Маршрут вебинара

Обработка ошибок

Нюансы

Обработка ошибок

Обработка ошибок для команд INSERT

<https://aristov.tech>

Например, обработка исключений помогает выполнить либо команду UPDATE, либо INSERT, в зависимости от ситуации.

Однако в современных приложениях вместо этого приёма рекомендуется использовать INSERT с ON CONFLICT DO UPDATE.

<https://www.postgresqltutorial.com/postgresql-tutorial/postgresql-upsert/>

Получение информации об ошибке

При обработке исключений часто бывает необходимым получить детальную информацию о произошедшей ошибке. Для этого в PL/pgSQL есть два способа: использование специальных переменных и команда GET STACKED DIAGNOSTICS.

Внутри секции EXCEPTION специальная переменная SQLSTATE содержит код ошибки, для которой было вызвано исключение. Специальная переменная SQLERRM содержит сообщение об ошибке, связанное с исключением. Эти переменные являются неопределёнными вне секции EXCEPTION.

Также в обработчике исключения можно получить информацию о текущем исключении командой GET STACKED DIAGNOSTICS, которая имеет вид:

GET STACKED DIAGNOSTICS *переменная* { = | := } *элемент* [, ...];

Каждый *элемент* представляется ключевым словом, указывающим, какое значение состояния нужно присвоить заданной *переменной*.

<https://www.postgresql.org/docs/current/plpgsql-control-structures.html#PLPGSQL-EXCEPTION-DIAGNOSTICS-VALUES>

Получение информации о месте возникновения ошибки

Команда GET DIAGNOSTICS, получает информацию о текущем состоянии выполнения кода (тогда как команда GET STACKED DIAGNOSTICS, рассмотренная ранее, выдаёт информацию о состоянии выполнения в момент предыдущей ошибки). Её элемент состояния PG_CONTEXT позволяет определить текущее место выполнения кода. PG_CONTEXT возвращает текст с несколькими строками, описывающий стек вызова. В первой строке отмечается текущая функция и выполняемая в данный момент команда GET DIAGNOSTICS, а во второй и последующих строках отмечаются функции выше по стеку вызовов.

<https://www.postgresql.org/docs/current/plpgsql-statements.html#PLPGSQL-STATEMENTS-DIAGNOSTICS>

Exception пример

```
DECLARE
  text_var1 text;
  text_var2 text;
  text_var3 text;
BEGIN
  -- здесь происходит обработка, которая может вызвать исключение
  ...
  EXCEPTION WHEN OTHERS THEN
    GET STACKED DIAGNOSTICS text_var1 = MESSAGE_TEXT,
                          text_var2 = PG_EXCEPTION_DETAIL,
                          text_var3 = PG_EXCEPTION_HINT;
END;
```

Потом например записать в таблицу с логами проблему. Или сделать raise notice - вывести сообщение в консоль. И если повысить детализацию системы логирования, эта информация также попадет в журнал.

Обработка ошибок

При выполнении команд в секции EXCEPTION локальные переменные функции на PL/pgSQL сохраняют те значения, которые были на момент возникновения ошибки. Однако, будут отменены все изменения в базе данных, выполненные в блоке

Наличие секции EXCEPTION значительно увеличивает накладные расходы на вход/выход из блока, поэтому не используйте EXCEPTION без надобности.

*Наличие секции EXCEPTION не позволит создавать транзакции **внутри блока**.*

Практика

Итоги

Итоги

Остались ли вопросы?

Увидимся на следующем занятии

Спасибо за внимание!

Когда дальше и куда?

Аристов Евгений