Базы данных Программирование баз данных

Многотабличные запросы, работа с результатами запросов. Оконные функции и представления. Модуль Psycopg2

Исходные таблицы

Таблица Pipelines

pipeline_name	max_diameter	length	year	country_id
Trans-Alaska	1220	1288	1977	1
Druzhba	1020	8900	1974	3
Kirkuk-Ceyhan	1170	970	1970	5
Hassi-Messaoud	720	805	1965	4
Seaway	762	1080	1976	1
Kolmogory-Klin	1220	2430	1985	3
Eastern Siberia-Pacific Ocean	1200	4740	2012	3
Enbridge	1220	6363	1950	2
Spearhead	610	1050	2006	1

Таблица Countries

id	country	overall_length
1	USA	4023360
2	Canada	840000
3	Russia	259913
4	Algeria	29642
5	Iraq	10437

Таблица Ratings

id	top_10
2	3
1	1
3	2
7	8
12	6

Таблица Employees

worker_id	last_name	salary	boss_id
1	Aksyutin	1500	3
2	Belousov	800	6
3	Miller	2000	
4	Kalinin	1000	6
5	Burmistrova	1100	3
6	Sechin	1600	
7	Markelov	1300	3

Работа с результатами запроса

SELECT columnList1 FROM TableName1
{UNION [ALL]|INTERSECT|EXCEPT}
SELECT columnList2 FROM TableName2 ...

1. UNION – объединение результатов запросов
2. INTERSECT – пересечение результатов запросов
3. EXCEPT – разность результатов запросов

SQL-запрос:
SELECT id FROM Ratings EXCEPT SELECT id FROM Countries

id
7
12

«Простое» объединение таблиц

SQL-запрос:
SELECT pipeline_name, country FROM Pipelines, Countries WHERE length > 4000 AND Pipelines.country_id = Countries.id

pipeline_name	country
Enbridge	Canada
Eastern Siberia-Pacific Ocean	Russia
Druzhba	Russia

Оператор JOIN – не простое объединение

SELECT columnList FROM TableName1
[INNER] | [[LEFT | RIGHT | FULL][OUTER]] JOIN TableName2
{[ON conditionList1| USING (columnList1)]}
[[INNER] | [[LEFT | RIGHT | FULL][OUTER]] JOIN TableNameN
{[ON conditionListN] | [USING (columnListN)]}

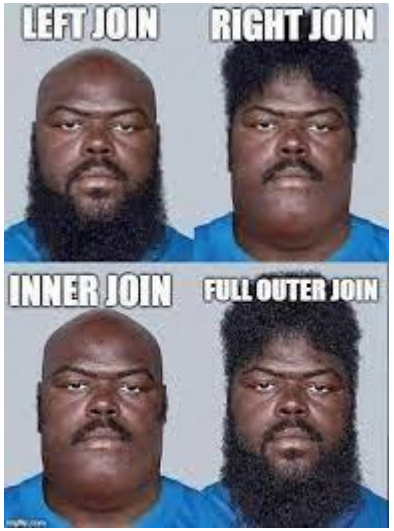
JOIN или INNER JOIN – внутреннее присоединение
Это такое соединение двух таблиц, при котором каждая запись из первой таблицы соединяется с каждой записью второй таблицы, создавая тем самым все возможные комбинации записей обеих таблиц (декартово произведение).

SQL-запрос:

```
SELECT pipeline_name, max_diameter, length, year, country_id, top_10
FROM Pipelines JOIN Ratings ON Pipelines.country_id = Ratings.id
```

pipeline_name	max_diameter	length	year	country_id	top_10
Spearhead	610	1050	2006	1	1
Seaway	762	1080	1976	1	1
Trans-Alaska	1220	1288	1977	1	1
Enbridge	1220	6363	1950	2	2
Kolmogory-Klin	1220	2430	1985	3	3
Eastern Siberia-Pacific Ocean	1200	4740	2012	3	2
Druzhba	1020	8900	1974	3	2

Оператор JOIN



OUTER JOIN – внешнее присоединение
Это соединение, которое включает в себя результаты запроса INNER с добавлением «неиспользованным» строк из одной из таблиц.
Перед JOIN указывается одно из ключевых слов LEFT, RIGHT или FULL, которые и определяют тип соединения:

1) LEFT JOIN – внешнее левое присоединение

Это такое соединение, которое возвращает все значения из левой таблицы, соединенные с соответствующими значениями из правой таблицы если они удовлетворяют условию соединения, или заменяет их на NULL в обратном случае.

2) RIGHT JOIN – внешнее правое присоединение. По аналогии с левым.

3) FULL JOIN – внешнее полное присоединение

Это такое соединение, которое выполняет внутреннее соединение записей и дополняет их левым внешним соединением и правым внешним соединением.

Оператор JOIN

SQL-запрос:

SELECT pipeline_name, max_diameter, length, year, country_id, top_10
FROM Pipelines LEFT JOIN Ratings ON Pipelines.country_id = Ratings.id

pipeline_name	max_diameter	length	year	country_id	top_10
Spearhead	610	1050	2006	1	1
Seaway	762	1080	1976	1	1
Trans-Alaska	1220	1288	1977	1	1
Enbridge	1220	6363	1950	2	2
Kolmogory-Klin	1220	2430	1985	3	3
Eastern Siberia-Pacific Ocean	1200	4740	2012	3	2
Druzhba	1020	8900	1974	3	2
Hassi-Messaoud	720	805	1965	4	NULL
Kirkuk-Ceyhan	1170	970	1970	5	NULL

SQL-запрос:

SELECT Ratings.id, top_10, Countries.id, country, overall_length FROM Ratings FULL JOIN Countries ON Countries.id = Ratings.id

id	top_10	id	country	overall_length	id
1	1	1	USA	4023360	1
2	3	2	Canada	840000	2
3	2	3	Russia	259913	3
NULL	NULL	4	Algeria	29642	NULL
NULL	NULL	5	Iraq	10437	NULL
7	8	NULL	NULL	NULL	NULL
12	6	NULL	NULL	NULL	NULL

Оператор JOIN

CROSS JOIN - перекрестное присоединение

Это такое соединение, при использовании которого, запрос выводит декартово произведение строк выбранных столбцов из двух таблиц. Нельзя задать условие после ключевого слова ON, поскольку по определению этот оператор соединяет каждую строку с каждой всеми возможными способами.

NATURAL JOIN – естественное присоединение

При использовании этого оператора создается неявное объединение на основе одинаковых имен столбцов в объединяемых таблицах. NATURAL JOIN по своим функциям сильно похож на INNER JOIN.

SQL-запрос:

SELECT * FROM Countries NATURAL JOIN Ratings

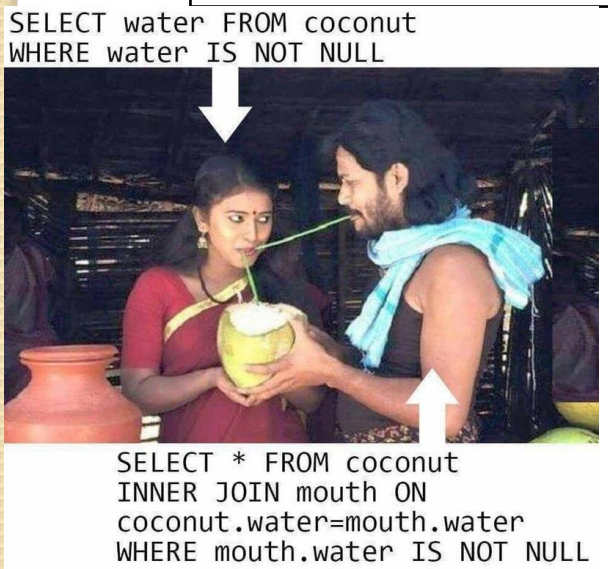
id	country	overall_length	top_10
1	USA	4023360	1
2	Canada	840000	3
3	Russia	259913	2

Оператор JOIN

SQL-запрос:

SELECT pipeline_name, year, country, top_10 FROM Countries JOIN Ratings ON Countries.id = Ratings.id JOIN Pipelines ON Ratings.id = country_id

pipeline_name	year	country	top_10
Spearhead	2006	USA	1
Seaway	1976	USA	1
Trans-Alaska	1977	USA	1
Enbridge	1950	Canada	3
Kolmogory-Klin	1985	Russia	2
Eastern Siberia-Pacific Ocean	2012	Russia	2
Druzhba	1974	Russia	2



Самоприсоединение таблиц

SQL-запрос:

```
SELECT e1.last_name as name, e1.salary, e2.last_name as boss_name,
e2.salary as boss_salary FROM Employees e1 LEFT JOIN Employees e2
ON e1.boss_id = e2.worker_id
```

name	salary	boss_name	boss_salary
Aksyutin	1500	Miller	2000
Burmistrova	1100	Miller	2000
Markelov	1300	Miller	2000
Kalinin	1000	Sechin	1600
Belousov	800	Sechin	1600
Sechin	1600	NULL	NULL
Miller	2000	NULL	NULL

Оконные функции

- ✔ Необходимо произвести вычисление над заданным набором строк, объединенных каким-то одним признаком.
- ✔ Можно сравнить с агрегатными функциями, но, в отличие от обычной агрегатной функции, при использовании оконной функции несколько строк не группируются в одну, а продолжают существовать отдельно.
- ✔ Результаты работы оконных функций просто добавляются к результирующей выборке как еще одно поле.

Оконные функции

SELECT column_names, function_name (column_for_calculations)
OVER ([PARTITION BY column_to_group_on]
[ORDER BY column_to_sort [ASC | DESC]]
[ROWS or RANGE expression_to_limit_the_rows_within_a_group])
FROM Table

column_names – имена столбцов, которые нужно вывести;
function_name – имя функции;
column_for_calculations – столбец, к которой применяется функция;
column_to_group_on – имена столбцов, по которым идет группировка;
column_to_sort - имена столбцов, по которым идет сортировка;
expression_to_limit_the_rows_within_a_group – выражение для ограничения строк в пределах группы;
Table – имя используемой таблицы.

PARTITION BY определяет столбцы, по которым будет производиться группировка, ORDER BY определяет столбцы, по которым будет проходить сортировка, необязательные инструкции ROWS или RANGE позволяют ограничить количество строк в окне.

Оконные функции

Таблица Oilfields

country	name_oilfield	reserves
Саудовская Аравия	Берри	1055
Кувейт	Большой Бурган	9100
Россия	Приобское	5000
Саудовская Аравия	Абкайк	12100
Саудовская Аравия	Хурайс	597
Россия	Самотлорское	7100
Саудовская Аравия	Эль-Катиф	487
Россия	Ромашкинское	5000
Кувейт	Сабрия	548

SQL-запрос:

```
SELECT country, name_oilfield, reserves, SUM(reserves) OVER() AS sum
FROM Oilfields
```

country	name_oilfield	reserves	sum
Саудовская Аравия	Берри	1055	40987
Кувейт	Большой Бурган	9100	40987
Россия	Приобское	5000	40987
Саудовская Аравия	Абкайк	12100	40987
Саудовская Аравия	Хурайс	597	40987
Россия	Самотлорское	7100	40987
Саудовская Аравия	Эль-Катиф	487	40987
Россия	Ромашкинское	5000	40987
Кувейт	Сабрия	548	40987

Оконные функции

Таблица Oilfields

country	name_oilfield	reserves
Саудовская Аравия	Берри	1055
Кувейт	Большой Бурган	9100
Россия	Приобское	5000
Саудовская Аравия	Абкайк	12100
Саудовская Аравия	Хурайс	597
Россия	Самотлорское	7100
Саудовская Аравия	Эль-Катиф	487
Россия	Ромашкинское	5000
Кувейт	Сабрия	548

SQL-запрос:

```
SELECT country, name_oilfield, reserves, SUM(reserves)
OVER(PARTITION BY country) AS Sum FROM Oilfields
```

country	name_oilfield	reserves	sum
Кувейт	Большой Бурган	9100	9648
Кувейт	Сабрия	548	9648
Россия	Самотлорское	7100	17100
Россия	Ромашкинское	5000	17100
Россия	Приобское	5000	17100
Саудовская Аравия	Хурайс	597	14239
Саудовская Аравия	Абкайк	12100	14239
Саудовская Аравия	Берри	1055	14239
Саудовская Аравия	Эль-Катиф	487	14239

Оконные функции

Таблица Oilfields

country	name_oilfield	reserves
Саудовская Аравия	Берри	1055
Кувейт	Большой Бурган	9100
Россия	Приобское	5000
Саудовская Аравия	Абкайк	12100
Саудовская Аравия	Хурайс	597
Россия	Самотлорское	7100
Саудовская Аравия	Эль-Катиф	487
Россия	Ромашкинское	5000
Кувейт	Сабрия	548

SQL-запрос:

```
SELECT country, name_oilfield, reserves, SUM(reserves)
OVER(PARTITION BY country ORDER BY name_oilfield ASC) AS sum
FROM Oilfields
```

country	name_oilfield	reserves	sum
Кувейт	Большой Бурган	9100	9100
Кувейт	Сабрия	548	9648
Россия	Приобское	5000	5000
Россия	Ромашкинское	5000	10000
Россия	Самотлорское	7100	17100
Саудовская Аравия	Абкайк	12100	12100
Саудовская Аравия	Берри	1055	13155
Саудовская Аравия	Хурайс	597	13752
Саудовская Аравия	Эль-Катиф	487	14239

Оконные функции

Виды функций:

Агрегатные функции – это функции, которые выполняют на наборе данных арифметические вычисления и возвращают итоговое значение:

- SUM – возвращает сумму значений в столбце;
- COUNT – вычисляет количество значений в столбце (значения NULL не учитываются);
- AVG – определяет среднее значение в столбце;
- MAX – определяет максимальное значение в столбце;
- MIN – определяет минимальное значение в столбце.

Оконные функции

Ранжирующие функции:

- ROW_NUMBER – функция возвращает номер строки и используется для нумерации;
- RANK – функция возвращает ранг каждой строки. В данном случае значения уже анализируются и, в случае нахождения одинаковых, возвращает одинаковый ранг с пропуском следующего значения;
- DENSE_RANK – функция возвращает ранг каждой строки. Но в отличие от функции RANK, она для одинаковых значений возвращает ранг, не пропуская следующий;
- NTILE – это функция, которая позволяет определить к какой группе относится текущая строка. Количество групп задается в скобках.

Оконные функции

country	name_oilfield	reserves
Саудовская Аравия	Берри	1055
Кувейт	Большой Бурган	9100
Россия	Приобское	5000
Саудовская Аравия	Абкайк	12100
Саудовская Аравия	Хурайс	597
Россия	Самотлорское	7100
Саудовская Аравия	Эль-Катиф	487
Россия	Ромашкинское	5000
Кувейт	Сабрия	548

Таблица Oilfields

SQL-запрос:

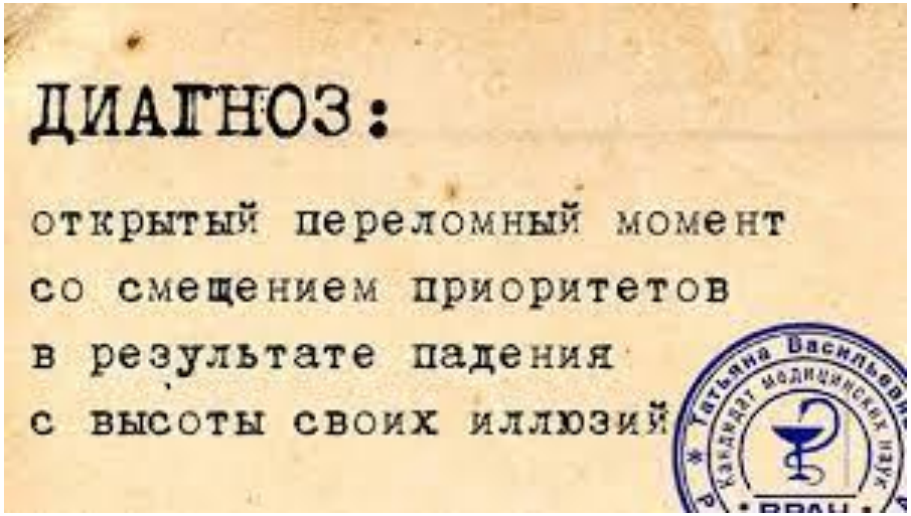
```
SELECT country, name_oilfield, reserves, ROW_NUMBER ()
OVER(PARTITION BY country ORDER BY reserves) AS row_number,
RANK() OVER(PARTITION BY country ORDER BY reserves) AS rank,
DENSE_RANK() OVER(PARTITION BY country ORDER BY reserves) AS
dense_rank, NTILE(3) OVER(PARTITION BY country ORDER BY
reserves) AS ntile FROM Oilfields
```

country	name_oilfield	reserves	row_number	rank	dense_rank	ntile
Кувейт	Сабрия	548	1	1	1	1
Кувейт	Большой Бурган	9100	2	2	2	2
Россия	Приобское	5000	1	1	1	1
Россия	Ромашкинское	5000	2	1	1	2
Россия	Самотлорское	7100	3	3	2	3
Саудовская Аравия	Эль-Катиф	487	1	1	1	1
Саудовская Аравия	Хурайс	597	2	2	2	1
Саудовская Аравия	Берри	1055	3	3	3	2
Саудовская Аравия	Абкайк	12100	4	4	4	3

Оконные функции

Функции смещения :

- LAG или LEAD – функция LAG обращается к данным из предыдущей строки окна, а LEAD к данным из следующей строки. Функцию можно использовать для того, чтобы сравнивать текущее значение строки с предыдущим или следующим. Имеет три параметра: столбец, значение которого необходимо вернуть, количество строк для смещения (по умолчанию 1), значение, которое необходимо вернуть, если после смещения возвращается значение NULL;
- FIRST_VALUE или LAST_VALUE – с помощью функции можно получить первое и последнее значение в окне. В качестве параметра принимает столбец, значение которого необходимо вернуть.



Оконные функции

country	name_oilfield	reserves
Саудовская Аравия	Берри	1055
Кувейт	Большой Бурган	9100
Россия	Приобское	5000
Саудовская Аравия	Абкайк	12100
Саудовская Аравия	Хурайс	597
Россия	Самотлорское	7100
Саудовская Аравия	Эль-Катиф	487
Россия	Ромашкинское	5000
Кувейт	Сабрия	548

Таблица Oilfields

SQL-запрос:

```
SELECT country, name_oilfield, reserves, LAG (reserves)
OVER(PARTITION BY country ORDER BY name_oilfield) AS lag, LEAD
(reserves) OVER(PARTITION BY country ORDER BY name_oilfield) AS
lead, FIRST_VALUE (reserves) OVER(PARTITION BY country ORDER
BY name_oilfield) AS first_v, LAST_VALUE (reserves) OVER(PARTITION
BY country ORDER BY name_oilfield) AS last_v FROM Oilfields
```

country	name_oilfield	reserves	lag	lead	first_v	last_v
Кувейт	Большой Бурган	9100	NULL	548	9100	9100
Кувейт	Сабрия	548	9100	NULL	9100	548
Россия	Приобское	5000	NULL	5000	5000	5000
Россия	Ромашкинское	5000	5000	7100	5000	5000
Россия	Самотлорское	7100	5000	NULL	5000	7100
Саудовская Аравия	Абкайк	12100	NULL	1055	12100	12100
Саудовская Аравия	Берри	1055	12100	597	12100	1055
Саудовская Аравия	Хурайс	597	1055	487	12100	597
Саудовская Аравия	Эль-Катиф	487	597	NULL	12100	487

Оконные функции

Аналитические функции:

- CUME_DIST – вычисляет интегральное распределение (относительное положение) значений в окне;
- PERCENT_RANK – вычисляет относительный ранг строки в окне.

SQL-запрос:

```
SELECT country, name_oilfield, reserves, CUME_DIST()
OVER(PARTITION BY country ORDER BY reserves) AS cume_dist,
PERCENT_RANK() OVER(PARTITION BY country ORDER BY reserves)
AS percent_rank FROM Oilfields
```

country	name_oilfield	reserves	cume_dist	percent_rank
Кувейт	Сабрия	548	0.5	0
Кувейт	Большой Бурган	9100	1	1
Россия	Приобское	5000	0.66	0
Россия	Ромашкинское	5000	0.66	0
Россия	Самотлорское	7100	1	1
Саудовская Аравия	Эль-Катиф	487	0.25	0
Саудовская Аравия	Хурайс	597	0.5	0.33
Саудовская Аравия	Берри	1055	0.75	0.66
Саудовская Аравия	Абкайк	12100	1	1

Представления

Объект, который содержит данные, полученные запросом SELECT из обычных таблиц.
Виртуальная таблица, к которой можно обратиться как к обычным таблицам и получить различные значения, которые в ней содержаться.
Может содержать в себе как данные из одной единственной таблицы, так и из нескольких таблиц.
Содержание представления повторяет выбранные данные из основных таблиц, но при этом представление не содержит никаких собственных данных.
Представления – подобия окон, через которые можно просматривать и использовать информацию, которая фактически хранится в базовой таблице.
Представление лишено физической материализации, поэтому указанный запрос будет выполняться при каждом обращении к представлению.

Создание и удаление представлений

CREATE [OR REPLACE] [TEMP | TEMPORARY] [RECURSIVE] VIEW name_view [(column_name_1[, ...])]
[WITH (view_parameter_name [= view_parameter_value] [, ...])] AS request
[WITH [CASCADED | LOCAL] CHECK OPTION]

name_view – название представления;
column_name_1 – имя столбца;
view_parameter_name – имя параметра представления;
view_parameter_value – значение параметра представления;
request – запрос.

DROP VIEW [IF EXISTS] name [, ...] [CASCADE | RESTRICT]

Опция OR REPLACE создает представление, но, если представление с этим именем уже существует, оно заменяется на новое. Новый запрос должен выдавать те же столбцы, что выдавал запрос, ранее определенный для этого представления, но можно добавить несколько новых столбцов в конце списка.

Создание и удаление представлений

Опция TEMPORARY создает временное представление, автоматически удаляется в конце работы.
Если создается временное представление с именем, которое уже занято постоянным представлением, то постоянное представление не будет видно, пока существует временное с таким же именем.
Чтобы обратиться к постоянному нужно либо удалить временное, либо в обращении дописать имя схемы.
Если в определении представления задействованы временные таблицы, представление так же создается как временное.
Параметр RECURSIVE создает рекурсивное представление. Для рекурсивного представления обязательно должен задаваться список с именами столбцов.
Параметр view_parameter_name и его значение view_parameter_value используются для защиты представлений от изменений (аналог CHECK OPTION).

Создание и удаление представлений

Опция [WITH [CASCADED | LOCAL] CHECK OPTION] создает представление с поверкой.
Если в результате выполнения INSERT или UPDATE в представлении исчезают строки, то такой запрос выполняться не будет. Проверка представления задается при помощи WITH CHECK OPTION.
Существующее представление должно быть удалено и создано заново для того, чтобы добавить в него проверку.
Будет производиться контроль, удовлетворяют ли новые строки условию, определяющему представление (то есть, проверяется, будут ли новые строки видны через это представление). Если они не удовлетворяют условию, операция не будет выполнена. Если указание CHECK OPTION отсутствует, команды INSERT и UPDATE смогут создавать в этом представлении строки, которые не будут видны в нем.
LOCAL – новые строки проверяются только по условиям, определенным непосредственно в самом представлении.
CASCADED – новые строки проверяются по условиям данного представления и всех нижележащих базовых .

Изменяемые представления

Изменяемые представления позволяют пользователям не только просматривать, но и редактировать данные.
Представление будет автоматически изменяемым, если оно удовлетворяют одновременно всем следующим условиям:

- Список FROM в запросе, определяющем представлении, должен содержать ровно один элемент, и это должна быть таблица или другое изменяемое представление;
- Определение представления не должно содержать предложения WITH, DISTINCT, GROUP BY, HAVING, LIMIT и OFFSET на верхнем уровне запроса;
- Определение представления не должно содержать операции с множествами (UNION, INTERSECT и EXCEPT) на верхнем уровне запроса;
- Список выборки в запросе не должен содержать агрегатные и оконные функции, а также функции, возвращающие множества.

Автоматически обновляемое представление может содержать как изменяемые, так и не изменяемые столбцы. Столбец будет изменяемым, если это простая ссылка на изменяемый столбец; в противном случае этот столбец будет доступен только для чтения.

Модуль *Psycopg2*

Начало работы:

```
pip install psycopg2
import psycopg2
```

Соединиться с СУБД:

```
conn = psycopg2.connect(database = "db1", user = "user1",
host="localhost", password = «12345»)
```

Здесь:

1)database – название базы данных, с которой будет работа;

2)user – имя пользователя;

3)host – адрес для подключения. Если вы работаете с локальной базой данных, то надо указывать «localhost»;

4)password – пароль пользователя;

5)conn – объект класса Connection. Обрабатывает соединение с экземпляром базы данных.

Создать курсор (объект класса cursor), указывающий на созданные выше соединение:

```
cur = conn.cursor()
```

Модуль *Psycopg2*

Чтобы выполнить запрос нужно написать следующие строки в программе:

```
cur.execute("SELECT * FROM people")
conn.commit()
```

Обратите внимание, что после вызова метода нужно подтвердить транзакцию методом commit(), методом соединения, а не курсора. Для отката к предыдущей транзакции нужно использовать метод rollback() того же соединения:

```
conn.rollback()
```

Имеются различные методы класса курсора.

По завершению работы нужно закрыть курсор и соединение с СУБД:

```
cur.close()
conn. close()
```

Правилом хорошего тона будет использование обработки исключений

Не нужно забывать про так называемые **«SQL-**инъекции**»**

The End