

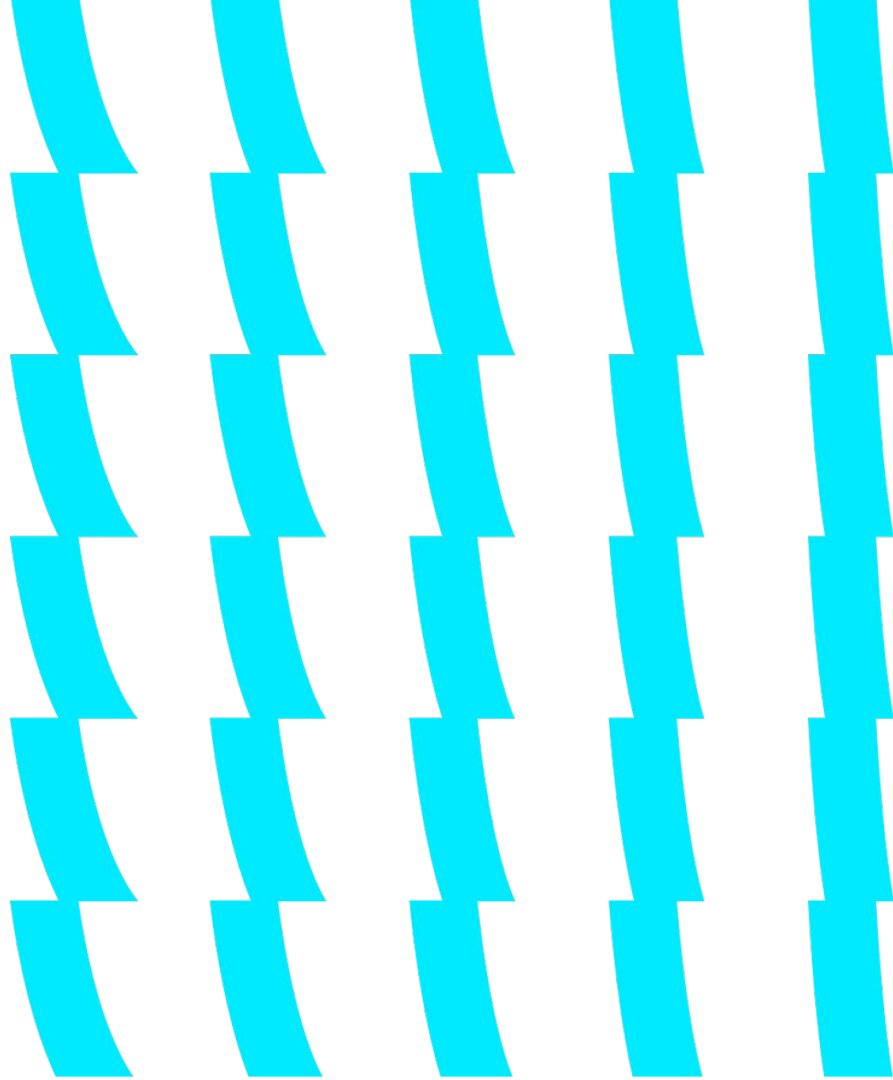
Автоматизация тестирования на Python

Юрьева Ольга



образование

**Не забудьте
отметиться на
портале**



Тестирование backend: SQL

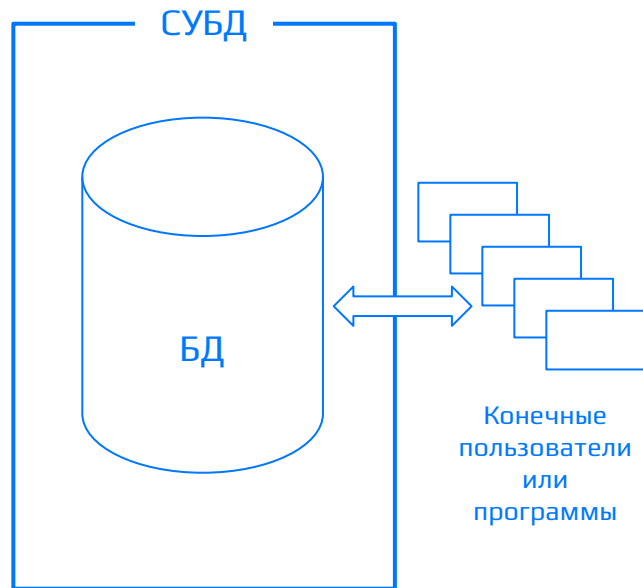


образование

Определение

База данных - данные хранимые в соответствии с правилами модели данных

Система управления базами данных (СУБД) — комплекс программ, позволяющих создать базу данных (БД) и манипулировать данными (вставлять, обновлять, удалять и выбирать). Система обеспечивает безопасность, надёжность хранения и целостность данных, а также предоставляет средства для администрирования БД



Модели БД

Реляционная



ORACLE®

NoSQL

Документо-
ориентированная



Распределенная



Ключ — значение



Кортеж



Колоночная



Временные ряды



SQL – Structured Query Language

DML

язык управления
данными

- SELECT
- INSERT
- UPDATE
- DELETE
- TRUNCATE
- EXPLAIN
- LOCK

DDL

язык описания данных

- CREATE
- ALTER
- DROP

DCL

язык управления
данными

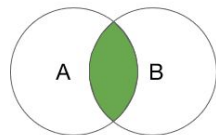
- GRANT
- REVOKE
- DENY

TCL

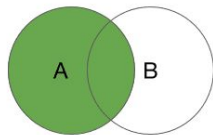
язык управления
транзакциями

- COMMIT
- ROLLBACK
- SAVEPOINT

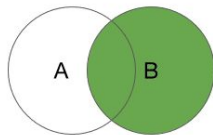
JOIN



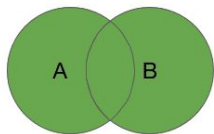
INNER JOIN



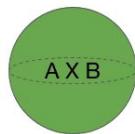
LEFT OUTER JOIN



RIGHT OUTER JOIN



FULL OUTER JOIN



CARTESIAN (CROSS) JOIN

- **INNER JOIN (JOIN)** — каждая строка из первой (левой) таблицы, сопоставляется с каждой строкой из второй (правой) таблицы, после чего, происходит проверка условия. В MySQL условие не обязательно, поэтому INNER JOIN стал аналогом CROSS JOIN. Если условия нет, или оно истинно, то строки попадают в результирующую таблицу.
- **LEFT JOIN (LEFT OUTER JOIN)** — важен порядок следования таблиц. Сначала происходит формирование таблицы соединением INNER JOIN. Затем, в результат добавляются записи левой таблицы, не вошедшие в результат после INNER JOIN. Для них, соответствующие записи из правой таблицы заполняются значениями NULL.
- **RIGHT JOIN (RIGHT OUTER JOIN)** — важен порядок следования таблиц. Аналогично LEFT JOIN, но во главе вторая таблица. Сначала происходит формирование таблицы соединением INNER JOIN. Затем, в результат добавляются записи правой таблицы, не вошедшие в результат после INNER JOIN. Для них, соответствующие записи из левой таблицы заполняются значениями NULL.
- **FULL JOIN (FULL OUTER JOIN)** — оператор FULL JOIN можно воспринимать как сочетание операторов INNER JOIN + LEFT JOIN + RIGHT JOIN. Сначала происходит формирование таблицы соединением INNER JOIN. Затем, в результат добавляются записи левой таблицы, не вошедшие прежде в результат. Для них, соответствующие записи из правой таблицы заполняются значениями NULL. Наконец, в таблицу добавляются значения не вошедшие в результат формирования из правой таблицы. Для них, соответствующие записи из левой таблицы заполняются значениями NULL.
- **CROSS JOIN** — каждая строка левой таблицы сопоставляется с каждой строкой правой таблицы. В результате получается таблица со всеми возможными сочетаниями строк обеих таблиц (*декартово произведение*).

Требования к реляционным БД

ACID

- Атомарность

Транзакция выполняется полностью или никак

- Согласованность

Сохранены могут быть только валидные данные

- Изолированность

Транзакции не влияют друг на друга (реализуется с помощью изоляции)

- Прочность

Зафиксированная транзакция не будет потеряна

Индексы

Объект, созданный для ускорения поиска и представляет собой упорядоченный набор пар значение и указатель на запись. Индексы могут быть реализованы различными структурами, но чаще это B деревья

```
create table `events` (  
  `timestamp` timestamp not null,  
  `date` date not null,  
  `banner_id` smallint(6) not null,  
  `event` char(20) not null,  
  `placement` char(50) not null,  
  primary key (`date`));
```

Транзакции и уровни изоляции

```
BEGIN;  
INSERT INTO test(id, value) VALUES (1, 'test'), (2, 'test 2');  
SELECT * FROM test;  
COMMIT;  
SELECT * FROM test;
```

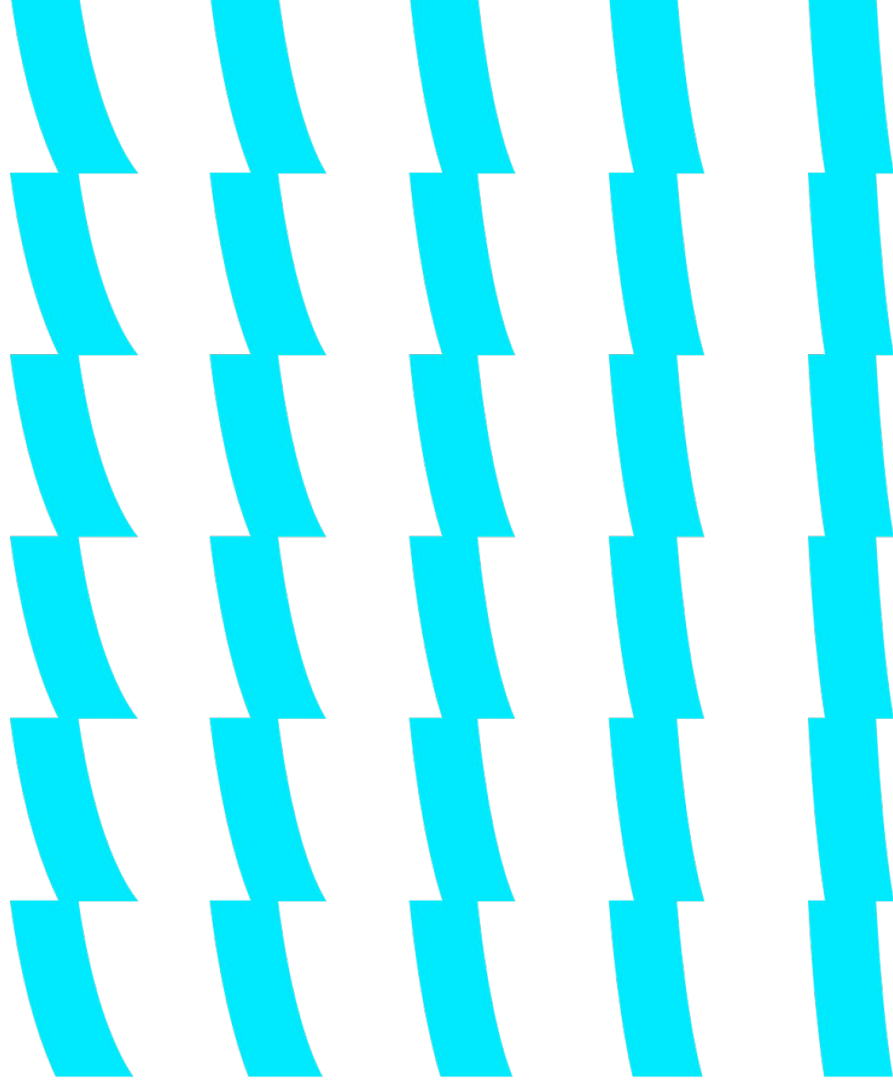
READ UNCOMMITTED - данные будут доступны сразу же после изменений все транзакции

READ COMMITTED - данные будут доступны только после коммита, причем до коммита они также будут недоступны внутри транзакции

REPEATABLE READ - данные доступны внутри транзакции сразу после изменения, но недоступны вовне до коммита

SERIALIZABLE - блокируется каждая строка над которой происходит какое либо действие

Большое спасибо!
Буду ждать ваших
ОТЗЫВОВ



Что почитать:

Введение в системы баз данных (К. Дж. Дейт)

Документация: <https://dev.mysql.com/doc/refman/8.0/en/>

<https://habr.com/ru/company/yandex/blog/522164/>

Oracle:

Oracle для профессионалов (Том Кайт)

<https://asktom.oracle.com/>