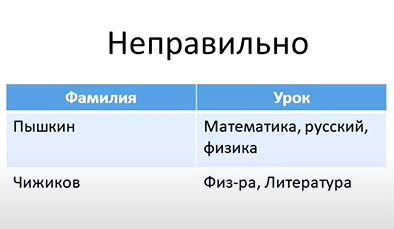
**Data Base**

**DB** – це сукупність ситематизованих даних, файл/файли на діску чи об’єкт пам’яті

**Реляційні бази даних** – це бд між записами якої існують відносини. Дані це таблиці, складері з ключа і значення. Реляційна бд – предсавляє собою набір зв’язаних між собою ключів

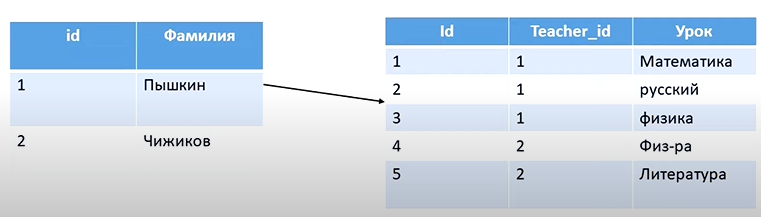
**Принципи побудови структури в реляційних бд:**

1. Одна колонка містить одне значення



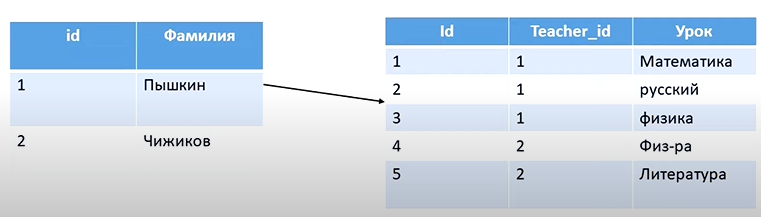


1. Кожен запис повинен містить унікальні ідентифікатори

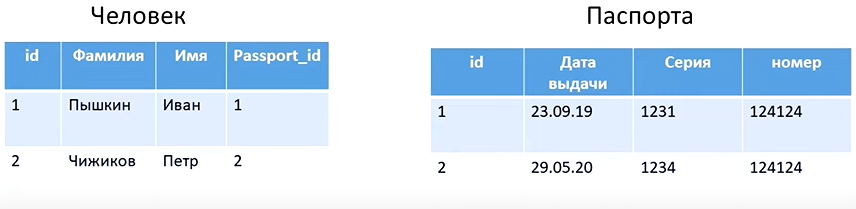


**Зв’язки в реляційних бд:**

1. Один до багатьох (one to many) – наприклад один вчитель може викладати декілька предметів. В даному випадку кожен предмет окрім свого id міститиме і id викладача, який його проводить, що і формує зв’язок



1. Один до одного (one to one) – в одної людини може бути лише 1 паспорт. В більшості випадків не потребує розподілу на таблиці, але для кращої семантики, дані розділяють і використовують id для прив’язки одних даних до інших



1. Багато до багатьох (many to many) – кожен студент може вчитись в багатьох викладачів і викладачі викладають багатьом студентам. В цьому випадку створюється окрема бд, в якій описані взаємозв’язки між вчителем - студентом (який студент вчиться в якого викладача і який викладач вчить якого студента)



SQL Commands:

**Команди створення полів таблиці:**

INSERT INTO table\_name (field\_name) value (“value”); – заповнення занчень полів таблиці

SELECT \* FROM table\_name; – виведення значень всіх полів таблиці (замість \* може бути перелічення або назва конкретного поля)

SELECT DISTINCT surname FROM table\_name; – виведення унікальних значень поля surname

SELECT \* FROM table\_name WHERE id=3 – виведе поле чи поля, що задовільняють вказану характеристику

SELECT \* FROM table\_name WHERE id < 5 LIMIT 2 – виведе поля, що задовільняють дану умову, але не більше вкзаного ліміту, **може вживатись і без умови**

SELECT id AS ‘ідентифікатор’, surname AS ‘Прізвище’ FROM table\_name – виведе таблиці, де змінить назви вказаних полів

SELECT \* FROM table\_name ORDER BY field\_name – виведе відсортовану таблицю

SELECT \* FROM table\_name ORDER BY field\_name DESC – виведе відсортовані значення в зворотньому порядку

ALTER TABLE table\_name ADD field\_name type\_of\_field – додає нове поле в таблицю

UPDATE table\_name SET field\_name = value WHERE id=1 – вставляє значення в вказане поле за вказаним id, без умови – вставить значення в дане поле у всіх

SELECT \* FROM table\_name WHERE field\_name LIKE “%ов” – поверне строки із полями, що закінчуються на ‘ов’

SELECT \* FROM table\_name WHER field\_name > 3 AND age < 45 – виведе строки, що містять стовбці, які задовільняють дані умови

SELECT \* FROM table\_name WHER field\_name > 3 OR age < 45 – виведе всі строки, які задовільняють одну із умов

SELECT \* FROM table\_name WHER NOT field\_name = 2 – виведе всі строки, які не відповідають даному значенню

SELECT \* FROM table\_name WHER age BETWEEN 25 AND 45 – виведе всі строки, які знаходяться в даному проміжку

DELETE FROM table\_name WHERE … - видалить строки, що підпадають даній умові

**Об’єднання таблиці:**

1. INNER JOIN / JOIN – в результуючу таблицю попадуть тільки ті значення, де для кожного вчителя є уроки



1. LEFT OUTER JOIN – потраплять всі вчителі, в незалежності від того чи ведуть вони уроки



1. RIGHT OUTER JOIN – потрапляють в вибірку всі уроки, навіть, якщо немає викладача, який за них відповідає



1. FULLJOIN – в вибірку потрапять всі значення з обох таблиць

**Команди об’єднання:**

SELECT teacher.surname, lesson.name FROM **INNER JOIN lesson ON teacher.id = lesson.teacher\_id –** відбувається злиття, предметів і вчителів, які взаємодіють і поєднані один з одним по id

SELECT teacher.surname, lesson.name FROM teacher LEFT OUTER JOIN lesson ON teacher.id = lesson.teacher\_id; - отримаємо навіть викладачів, які уроки не ведуть

SELECT teacher.surname, lesson.name FROM teacher RIGHT OUTER JOIN lesson ON teacher.id = lesson.teacher\_id; - отримаємо всі предметри, навіть, якщо їх ніхто не веде

SELECT \* FROM lesson UNION SELECT \* FROM teacher – об’єднання по вертикалі

Команди вичислення:

SELECT AVG(field\_name) FROM teacher – поверне середнє значення даного поля по всій таблиці

SELECT MAX(field\_name) FROM teacher – поверне максимальне значення даного поля по всій таблиці

SELECT MIN(field\_name) FROM teacher – поверне мінімальне значення даного поля по всій таблиці

SELECT SUM(field\_name) FROM teacher – поверне суму всіх значень даного поля в таблиці

SELECT age, COUNT(age) FROM teacher GROUP BY age – виведе таблицю, яка вкаже, к-сть дублів в таблиці даного значення age

**Реляційні БД:**

**MySQL**

<https://www.mysql.com/>, <https://www.mysql.com/products/community/> , <https://www.npmjs.com/package/mysql> , <https://www.npmjs.com/package/mysql2>

**MariaDB:**

<https://mariadb.org/> , <https://www.npmjs.com/package/mariadb> , <https://www.npmjs.com/package/mysql> , <https://www.npmjs.com/package/mysql2>

**SQLite:**

<https://www.sqlite.org/index.html> , <https://www.npmjs.com/package/sqlite3>

**Oracle:**

<https://www.oracle.com/database/>

**AWS:**

<https://aws.amazon.com/ru/rds/>

**CloudSQL:**

<https://cloud.google.com/sql>

**Azure SQL:**

<https://azure.microsoft.com/>

**Нереляційні бази даних** – всі інші бази даних, які не підпадають під опис реляційних баз даних

**Не реляційні БД:**

**Redis:**

<https://en.wikipedia.org/wiki/Redis>  , <https://www.npmjs.com/package/redis>

**Couchbase:**

<https://www.npmjs.com/package/couchbase>

**Riak:**

<https://www.npmjs.com/package/@bunchtogether/riak-client>

**Elasticsearch:**

<https://www.npmjs.com/package/elasticsearch>

**Apache\_Cassandra:**

<https://www.npmjs.com/package/cassandra-driver>

**AWS-SDK:**

<https://www.npmjs.com/package/aws-sdk>