What is a database?

Any collection of related information.

- Phone book
- To do list etc

They can be stored in many different ways.

eg. mind, paper, computer.

DBMS - Database management Systems

These help users create and maintain a DB.

C	R	U	D
Create	Read	Update	Delete

Two types of databases.

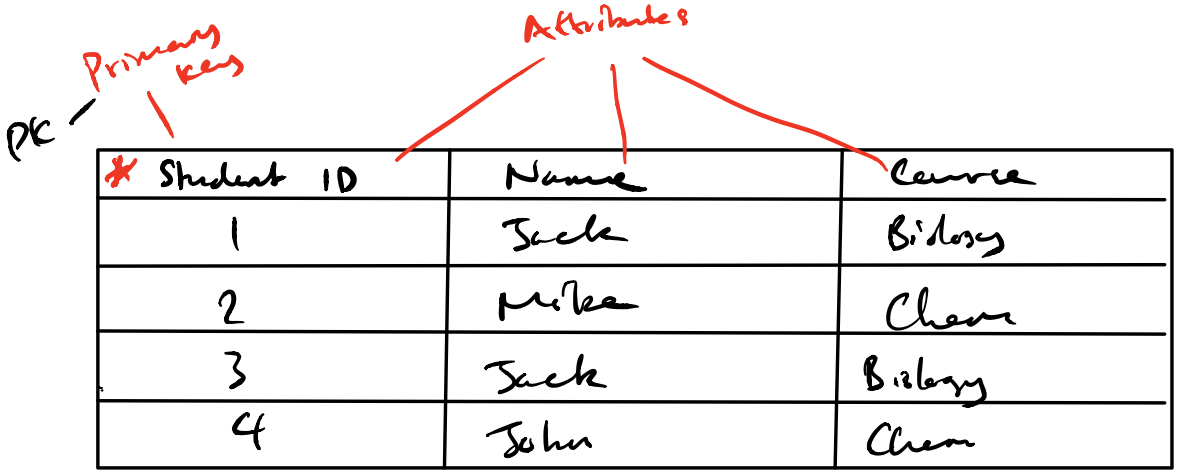
- Relational DB (SQL)
  Organise data into one or more tables.
- Non-relational
  Organise data is anything but a traditional table.

SQL - Structured Query Language.

Query - Request to the DB for specific info

Wrap Up

- Database is any collection of related information
- Computer are great for storing databases
- Database Management Systems (DBMS) make it easy to create, maintain and secure a database.
- DBMS allow you to perform the C.R.U.D operations and other administrative tasks
- Two types of Databases, Relational & Non-Relational
- Relational databases use SQL and store data in tables with rows and columns
- Non-Relational data store data using other data structures



. Natural key used as (PK) is something that has a mapping with real life. eg. passport number, NI number.

. Surrogate key. - Has no mapping eg. Student number.

. Foreign key. - Stores the pk of one entity into another entity.

. Composite key (CK) - Requires 2 attributes as PK. Together, they uniquely identify a table.
Can have a composite key which serves as a foreign key. 2 foreign keys together will become a CK.

Structured Query Language (SQL)

Queries

- Tells the RDBMS what information you want to retrieve.

```
SELECT * FROM students;
```

This will show you the table created.

```
Insert into students (name, course) Values (' ', '');
```

```
Update students
SET course = 'Chem'
Where course = 'Chemistry';
```

can be any attribute.

```
Delete
where = ' ';
```

Using Queries

(1)

Select * — All information

Select name - will print all names

(2)

Order by name ASC; (ascending)

(3)

(3) Select * first name, last name AS forename

Changes the attribute name

(4) Distinct - removes duplicates.

(5) Count

(6) Avg

(7) Sum

(8) Group by - Aggregate, can count seperately eg. Number of 'M' and 'F' receiving a salary.

```
Select count(salary), sex - This will add another column to results
From employee
Group by sex;
```

```
select sum(total_sales) ,emp_id
from works_with
group by emp_id;
```

Total sales of each employee

Success 7:15 AM
4 rows 0.162 seconds
Explore | SQL | Data | Chart | Export

sum(total_sales)	emp_id
282000	102
218000	105
31000	107
34500	108

Wild Cards

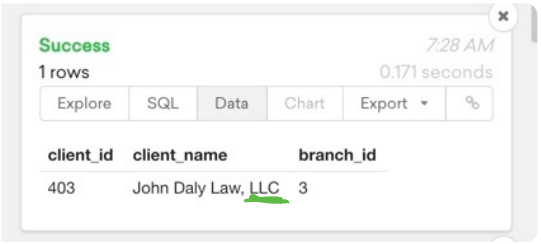
```
select *
from client
where client_name like '%LLC';
```

At end

%LLC%

By using Like & % sign, you can select any name which has the following letters

anywhere letters.

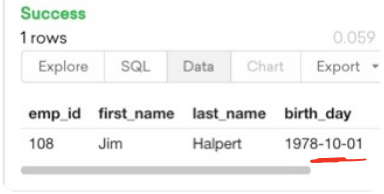


_ can be used for 1 letter

% Any number of Char

```sql
-- any employee born in october
select *
from employee
where birth_date like '____-10%';

-- use of 4 '_' for the year, as each one is for one characte
```



Union

```sql
-- to get a list of 2 sets of data.
-- you need to run each one individually.
select first_name
from employee;
select client_name
from client;
```

```sql
-- by adding union, you get both informations in one list.
select first_name
from employee
union
select client_name
from client;
-- you can only select same number of attributes.
-- the following code is incorrect.
select first_name,last_name
from employee
union
select client_name
from client;
```

2 attributes

only 1

Joins

```sql
insert into branch values(4, 'buffalo',null,null);
-- find all branches and the name of their managers
select employee.emp_id,employee.first_name,branch.branch_name
from employee
join branch
on employee.emp_id = branch.mgr_id;

-- When you use joins, you get info from different tables and present them into 1
-- in this example, the tables employee and branch were used.
-- both have a common coloumn which they share, therefore can use joins.
-- in the example, mgr_id and emp_id are shared.
```

Success 11:10 AM

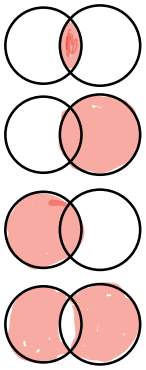
3 rows 0.102 seconds

Explore | SQL | Data | Chart | Export

emp_id	first_name	branch_name
100	David	Corporate
102	Michael	Scranton
106	Josh	Stamford

. There are 4 different types of joins.

- Inner join
- Right join
- Left join
- outer join



Left join

```sql

select employee.emp_id,employee.first_name,branch.branch_name
from employee
left join branch
on employee.emp_id = branch.mgr_id;

-- every employee will be added as it is the first table.

```

emp_id	first_name	branch_name
100	David	Corporate
101	Jan	NULL
102	Michael	Scranton
103	Angela	NULL
104	Kelly	NULL
105	Stanley	NULL
106	Josh	Stamford
107	Andy	NULL
108	Jim	NULL

Right join

```sql

select employee.emp_id,employee.first_name,branch.branch_name
from employee
right join branch
on employee.emp_id = branch.mgr_id;

-- every branch name will be added as it is the second table.

```

4 rows 0.225 seconds

Explore | SQL | Data | Chart | Export

emp_id	first_name	branch_name
100	David	Corporate
102	Michael	Scranton
106	Josh	Stamford
NULL	NULL	buffalo

Nested Queries

```sql
-- Nested queries
-- multiple 'select' statements to get information.

-- find all employees who have
-- sold over 30000 to a single client.

-- first select all employees who have sold over 30000.
-- select emp_id
-- from works_with
-- where total_sales > 300000

-- thereafter enter the data you want,
select employee.first_name, employee.last_name
from employee
where employee.emp_id in
(
-- open parenthesis, this is where i will nest
-- the info from the first query
select works_with.emp_id
from works_with
where works_with.total_sales > 30000
);
```

This info is nested in

CLEAR ALL

Success 11:40 AM

2 rows 0.067 seconds

Explore | SQL | Data | Chart | Export

first_name	last_name
Michael	Scott
Stanley	Hudson

On delete

```
ALTER TABLE employee
ADD FOREIGN KEY(branch_id)
REFERENCES branch(branch_id)
ON DELETE SET NULL;
```

- Once you delete that info, it will set it to null

ON delete Cascade - This will delete entire rows where it affects. Use this where deleting PK.

Triggers (3:30) stopped