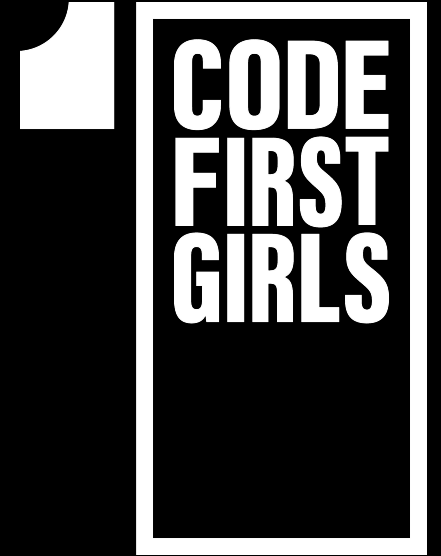
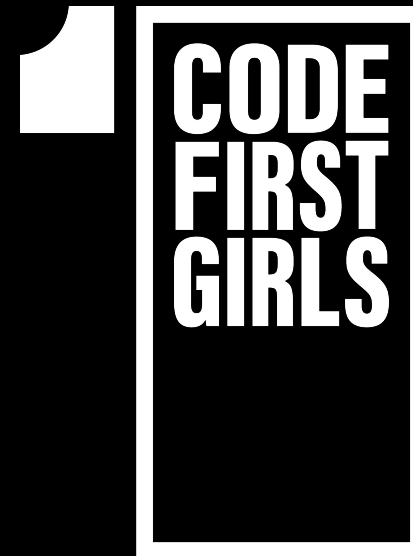


DATABASES & INTRODUCTION TO SQL

LESSON 1





WHAT IS BIG DATA?



INFORMATION

Big Data



HUGE VOLUME

Structured and Unstructured

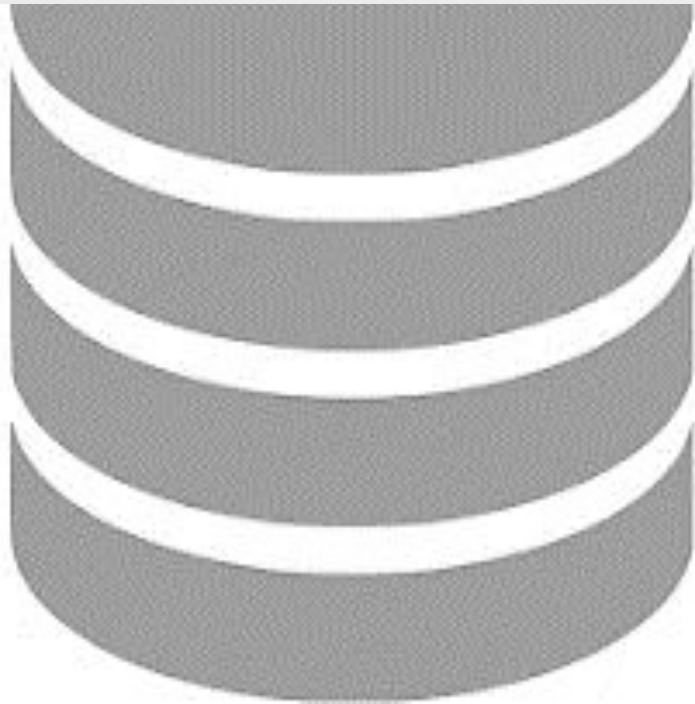


 **CAPTURED**
Data Collection

 **STORED**
Data Persisted

 **ANALYSED**
Data Study





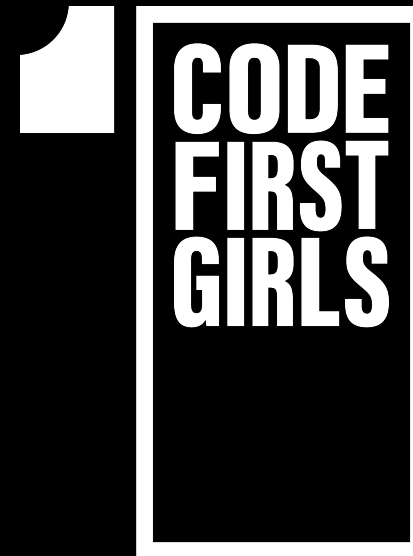
DATABASE

Data Storage

SQL PROGRAMMING

Data Access and Processing





HOW IS DATA USED IN TECH?

MOST COMMON APPLICATION FIELDS



SOFTWARE ENGINEERING

Programming and Development



DATA ANALYSIS

Data Science and Data Processing

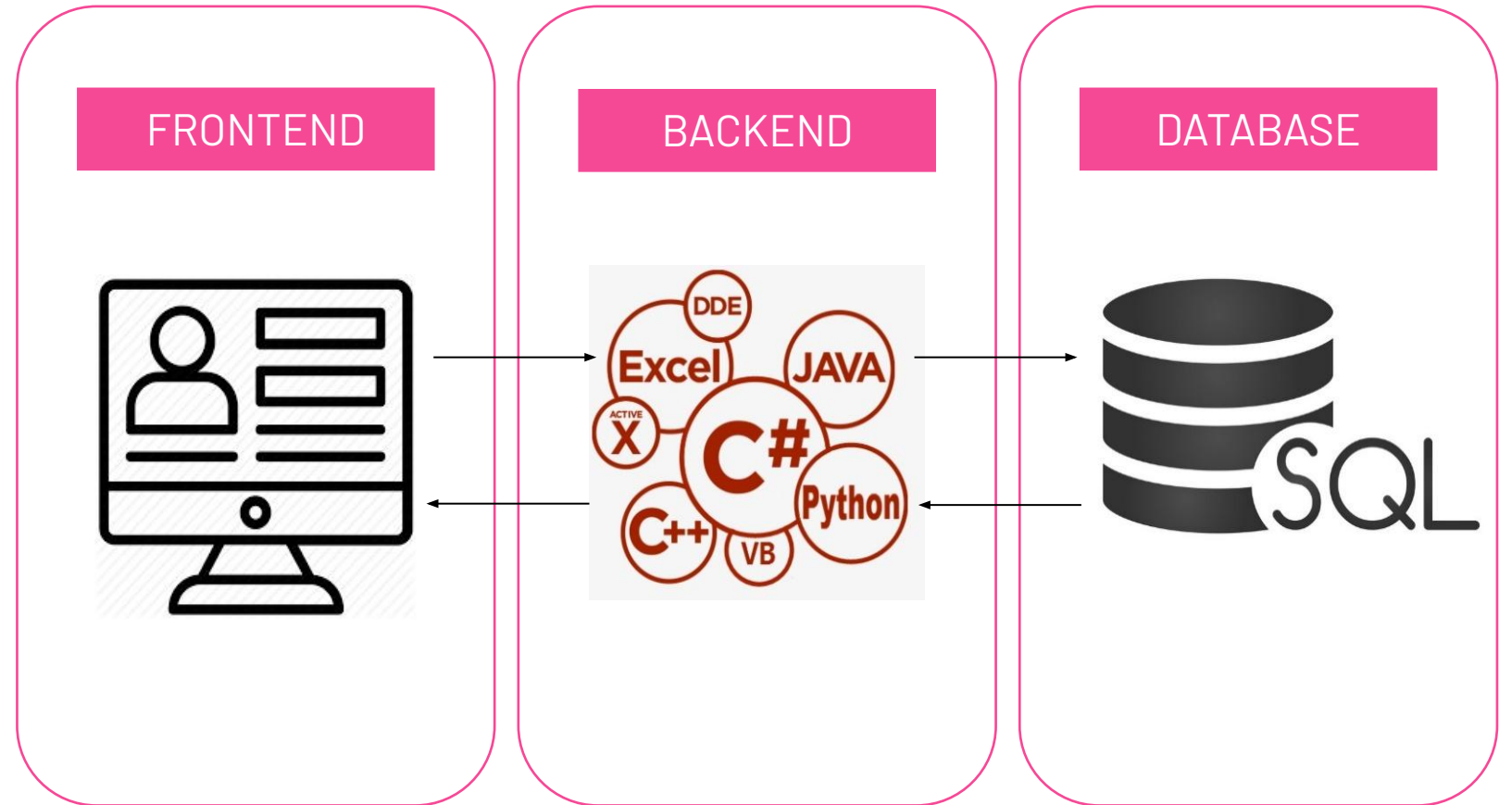




SOFTWARE ENGINEERING

Programming and Development

LAYERS MODEL

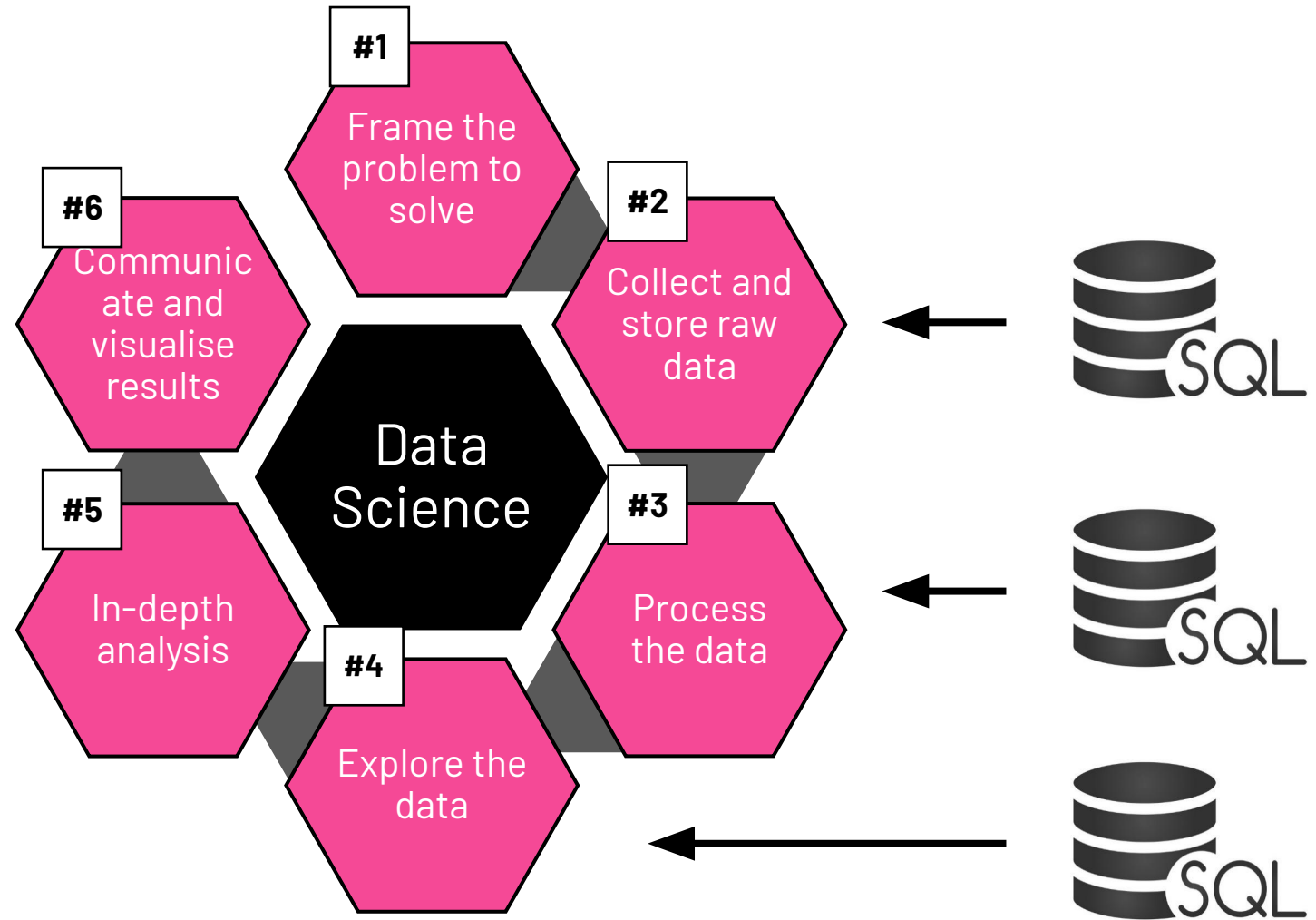




DATA ANALYSIS

Data Science and Data Processing

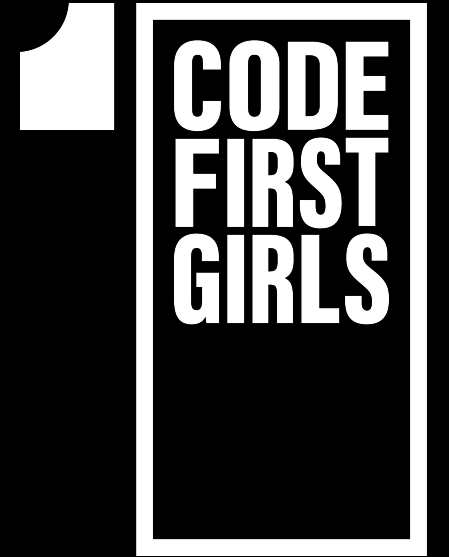
DATA SCIENCE STEPS



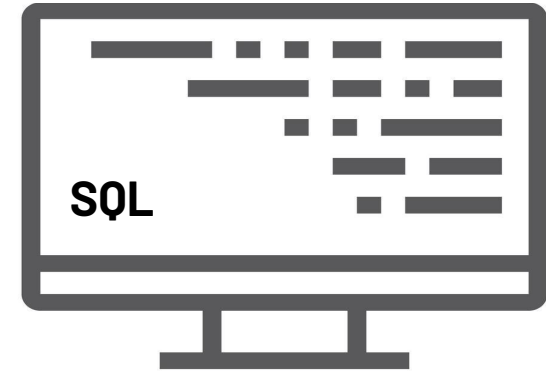
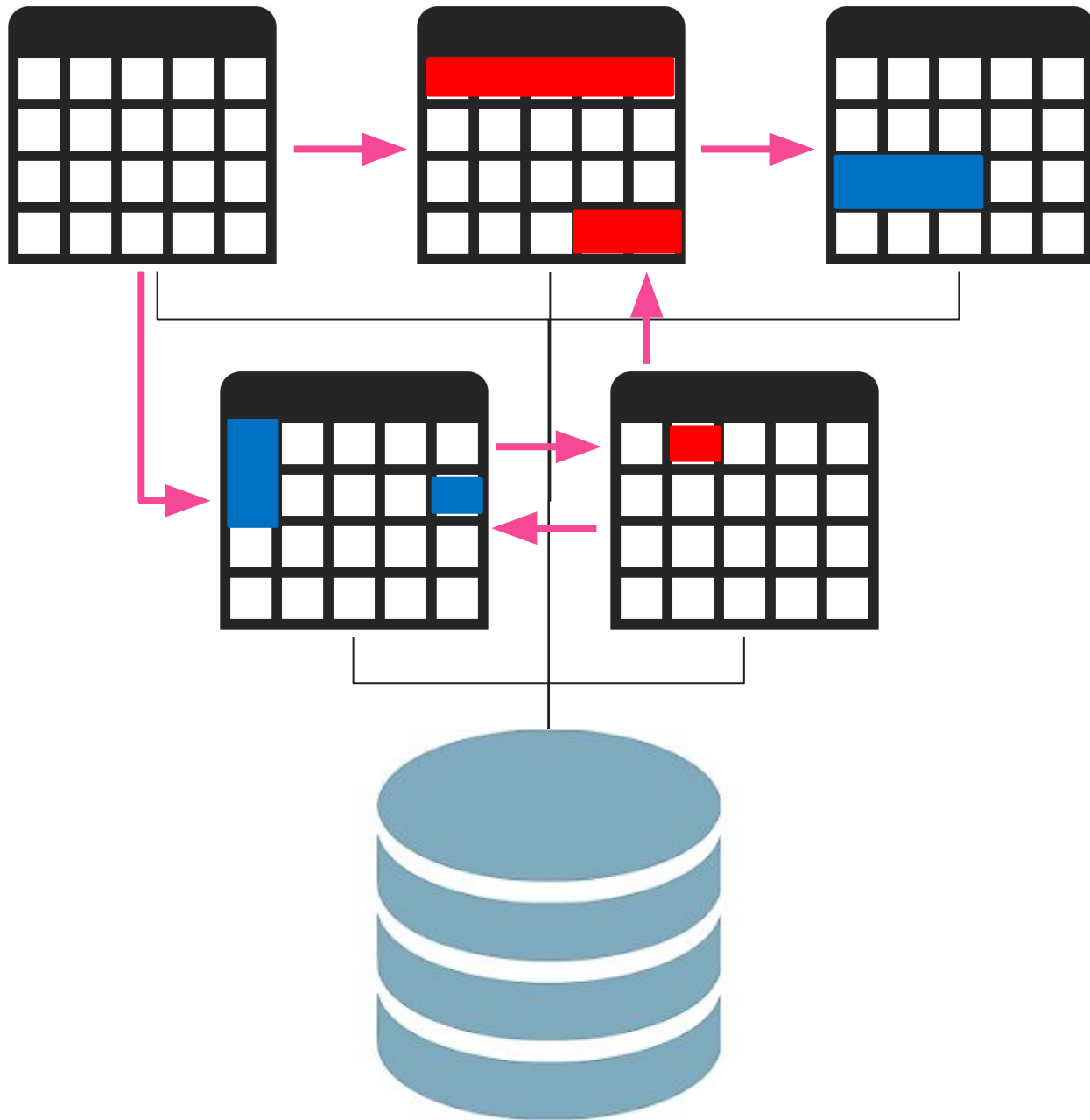
AGENDA



- 01** Understand what is SQL and Databases (DB) and how to use them
- 02** Learn about different types of data
- 03** Create our first Database and Tables
- 04** Learn core commands to insert and get data from a DB



WHAT IS A DATABASE?



SQL code to query the database to extract required data

Data sets combined within DB as per query to extract required data

Data extracted or can be analysed even further if required



RELATIONAL DATABASE MODEL (RDBM)

RDBM - a way to describe how we are going to store the data in the DB and how those individual pieces of data are going to contain relationships to each other

- In a relational database, we store data inside of something called a table
- Tables in a DB are related to one another. If we look at the DB diagram, we can see those relations depicted with lines/arrows
- From the diagram we can also see all table names, their columns and data types stored in each column

NB: read more about RDBM <https://www.codecademy.com/articles/what-is-rdbms-sql>



EXAMPLE – EER DIAGRAM

Let's review a Relational Database Diagram for a DB called "Company".

TABLE employees

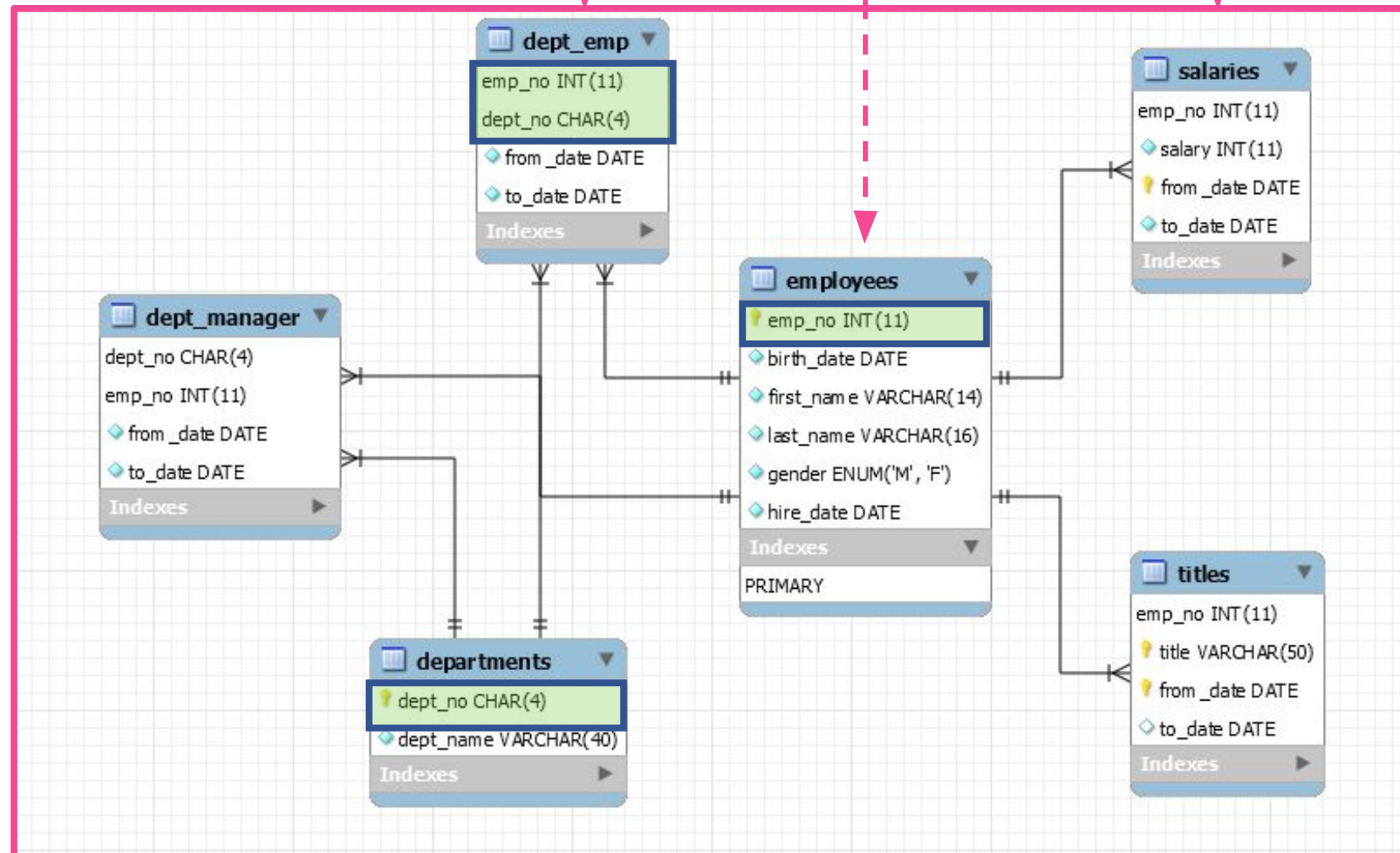
It has 6 columns and it is related to other tables

TABLE salaries

It is related to the table employees, because each employee has a band (salary ID) and salary (amount).

TABLE dept_emp

It is a JUNCTION table: because it has id combinations of employees and departments, i.e. which employee belongs to which department. It cannot exist without tables employee or departments.



ASK DB A QUESTION



- One of the most common tasks is to retrieve data from a DB. Let's imagine that we have a simple table in our DB that contains information about people

These are our contacts

Name	Surname	Telephone
Lucy	Smith	020 7777 8888
Mike	Peters	020 2222 3333
Julie	Andrews	074 7878 1212



How many of my contacts have a surname starting with 'P'?

- Asking DB questions is called 'QUERYING'
- Each query is a piece of code that we write in SQL language

TABLES COLUMNS NAMES



- When we create a DB and tables within that database, we need to specify each column type, i.e. what kind of data will this column hold
- **We also need to specify any constraints or restrictions that a column might have. It might be type of data, size of data, format of data or disclaimer where it is required or not**
- Compare the two tables below: the latter does not have a value in the telephone column, but has something else

Example 1

Name	Surname	Telephone
Lucy	Smith	020 7777 8888
Mike	Peters	020 2222 3333

Example 2

Name	Surname	Telephone
Lucy	Smith	null
Mike	Peters	020 2222 3333

SQL DATA TYPES (CORE)

- We will be using these data types to create tables and specify what kind of values each column should hold

DATA TYPE	VALUE SPACE
CHAR(size)	Fixed-length strings, where size is the number of characters to store. (max 255 characters)
VARCHAR(size)	Variable-length strings, where size is the number of characters to store. (max 255 characters)
BINARY	Fixed-length strings, where size is the number of binary characters to store. (max 255 characters)
FLOAT(p)	Floating point number, where p is a precision
INTEGER	Standard integer value, allows whole numbers
DECIMAL(m,d)	Fixed point number, where m is the total digits and d is the number of digits after the decimal.
BOOLEAN	Data type where a value of 0 is considered to be FALSE and any other value is considered to be TRUE
DATE	Displayed as 'YYYY-MM-DD'.
TIME	Displayed as 'HH:MM:SS'.
TIMESTAMP	Displayed as 'YYYY-MM-DD HH:MM:SS'.

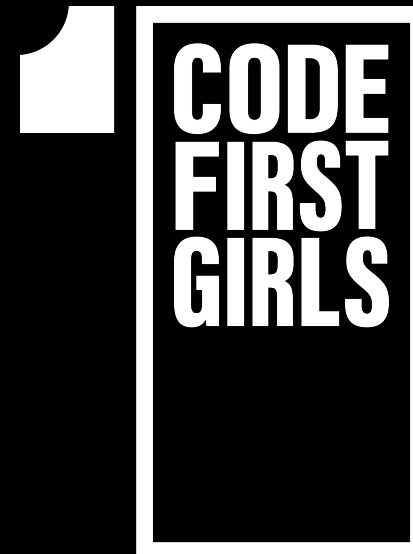
NB: read more about data types:

https://www.w3schools.com/sql/sql_datatypes.asp

NULL VALUES

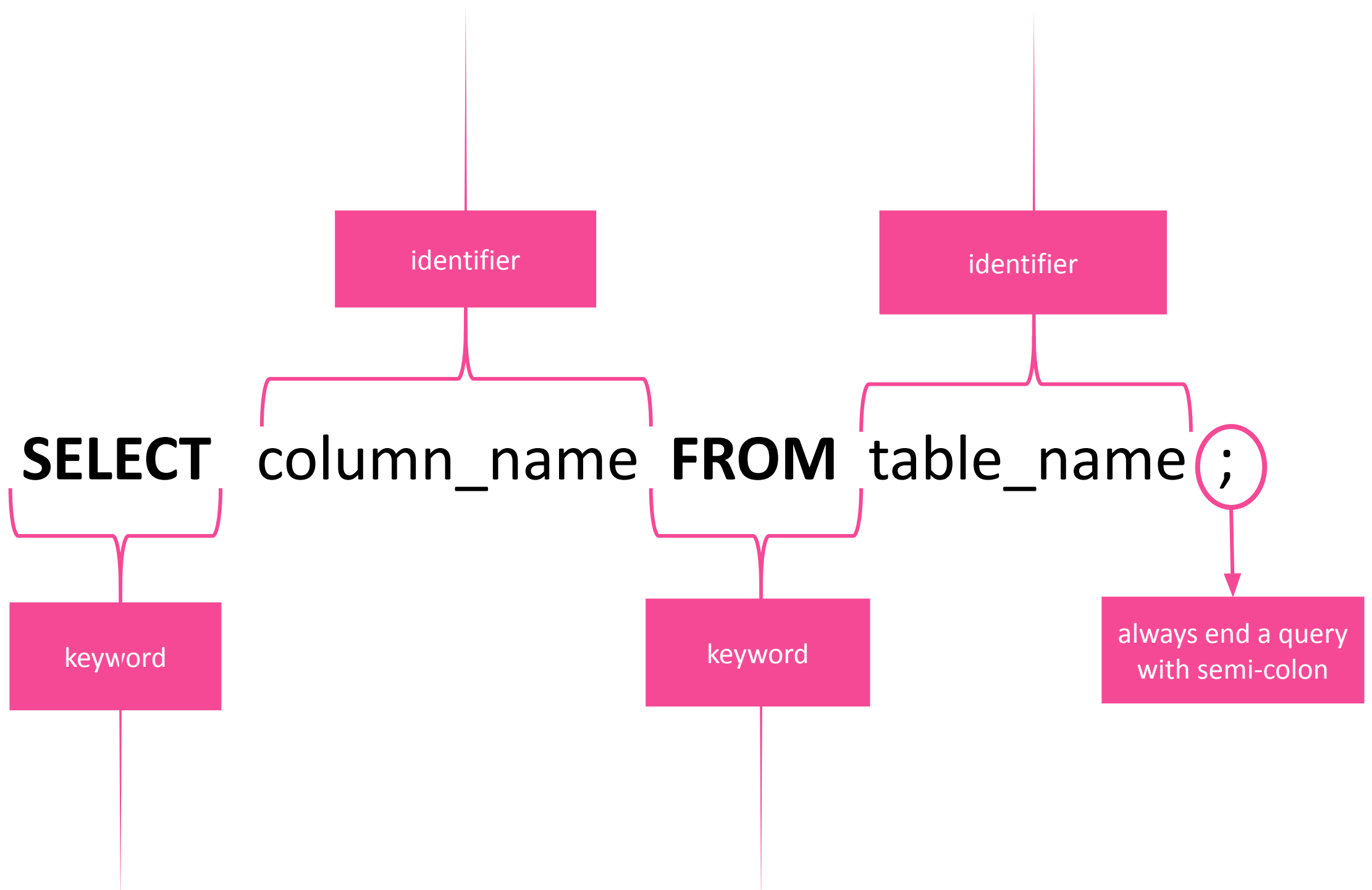
- **NULL** is a special value in SQL, it indicates a lack of a value!
- **NULL** value is different from a zero value or a field that contains spaces: NULL value is one that has been left blank during record creation (it means data does not exist in the database)
- Columns can be required or not required: required is **NOT NULL** and not required is **NULL**

NULL	NOT NULL
Default for a column definition	Must be specified on column definitions
It means that we are allowed to insert NULL values	It means we are not allowed to insert NULL values. Inserting a NULL would raise an error!



SQL SYNTAX

CODING QUERIES



SELECT

- Think of a **SELECT** statement as a question that we ask the database

EXAMPLE QUESTIONS

- What are all the fruit that we have in stock?
- How many contacts do we have in our phone book?
- Who are all my contacts with surname 'Smith'?
- How many flights to New York do we have on Tuesday?
- What is the party with the highest number of voter region?

SELECT (LIST)

- Quite often SELECT statement contains a list of columns that follow this key word
- The names of these columns come from a table that we are using for this query ('our question')
- We add a coma after each column name, except for the last column
- Then we need to include a FROM statement to specify the table that is required for this query



SYNTAX

```
SELECT  
<column_name>,  
<column_name>  
  
FROM <table_name>;
```

EXAMPLE

```
SELECT  
first_name,  
last_name  
FROM person;
```

Table Person

first_name	last_name	telephone
Lucy	Smith	020 7777 8888
Mike	Peters	020 2222 3333
Julie	Andrews	074 7878 1212



SYNTAX

The * character is a wildcard that gets ALL columns from a table. It also may be called 'select list' of all columns

```
SELECT *  
FROM <table_name>;
```

EXAMPLE

```
SELECT *  
FROM person;
```

BAD PRACTICE 😞

FROM

FROM clause specifies the table that we want to query, i.e. the table that we want to use when asking a database a question.

PLEASE NOTE

- SQL syntax also allows us to qualify a column name with a table name – it is very useful when we write complex queries that involve multiple tables.
- We can also alias a table name with any word or abbreviation – it helps us to avoid wordy statements and make our syntax neater.



SYNTAX

Column names are qualified with the
table name

```
SELECT  
<table_name>.<column_name>,  
<table_name>.<column_name>,  
  
FROM <table_name>;
```

EXAMPLE

```
SELECT  
person.first_name,  
person.last_name  
FROM person;
```

GOOD PRACTICE 😊



SYNTAX

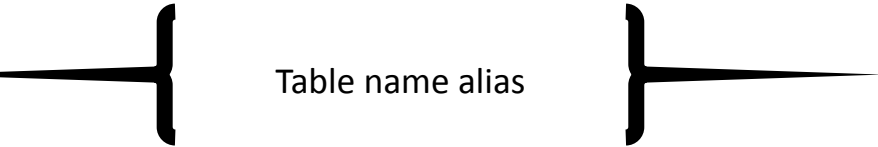


Table name alias

```
SELECT  
<alias>.<column_name>,  
<alias>.<column_name>,  
  
FROM <table_name> <alias>;
```

EXAMPLE

```
SELECT  
p.first_name,  
p.last_name  
FROM person p;
```

BEST PRACTICE 😊

PRACTICE



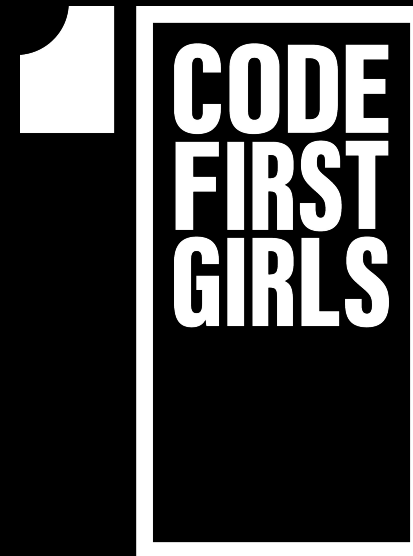
LET'S WRITE OUR FIRST QUERIES TOGETHER

<https://coderpad.io/sandbox>

```
Run ▶ Info MySQL ▼
1 /*
2 CoderPad provides a basic SQL sandbox with the following schema.
3 You can also use commands like `show tables` and `desc employees`
4
5 employees                                projects
6 +-----+-----+ <---+ +--> +-----+-----+
7 | id      | int    | <---+ +--> | id      | int    |
8 | first_name | varchar | | | | title   | varchar |
9 | last_name  | varchar | | | | start_date | date   |
```

WRITE THE FOLLOWING QUERIES

- Go to the sandbox MySQL and write a query to show me **all project names and their budgets.**
- Show me **all information from the employees** table
- Let's write our very first queries together!



SQL SYNTAX

BUILDING A DATABASE

CREATE DATABASE

- Oddly this statement not part of the SQL Standard
- It is supported by most implementations
- USE DATABASE command to scope future queries
- We can also fully qualify table name to database



SYNTAX

Example 1

CREATE DATABASE <DB_name>;

Example 2

USE DATABASE <DB_name>;

SELECT *

FROM <table_name>;

Example 3

SELECT *

FROM <DB_name>.<table_name>;

EXAMPLE

CREATE DATABASE Food;

☐ CREATE DB COMMAND

USE DATABASE Food;

SELECT *

☐ USE DB COMMAND

FROM Fruit;

☐ ALL QUERIES WILL BE IN THIS DB

SELECT *

FROM Food.Fruit

☐ FULLY QUALIFIED TABLE NAME

CREATE TABLE

- Is part of SQL Standard
- Followed by table name
- Then list of column definitions
- At minimum column name and type



SYNTAX

```
CREATE TABLE <table_name>  
(col1 Type,  
 col2 Type,  
 col3 Type);
```

EXAMPLE

```
CREATE TABLE customers  
(person_id INTEGER,  
 name VARCHAR(50),  
 surname VARCHAR(50),  
 telephone INTEGER);
```

☐ CREATE TABLE + TABLE NAME

☐ COLUMN NAMES AND TYPES



SYNTAX

```
CREATE TABLE <table_name>  
(col1 Type,  
 col2 Type,  
 col3 Type);
```

EXAMPLE

```
CREATE TABLE customers  
(person_id INTEGER NOT NULL,  
  name VARCHAR(50),  
  surname VARCHAR(50) NOT NULL,  
  telephone INTEGER);
```

- **NOT NULL** (these values are required)
- **NULL** (this value can be omitted)
- **NOT NULL** (must have)

PRACTICE



It is time to create our very first database and populate it with few tables.

TASKS

- Create a database called **Bakery**.
- Let's add two tables to the database. One should be called 'Sweet' and the one should be called 'Savoury'
- The Sweet table should have three columns: **id** (which is a number), **item_name** (name of a pastry) and **price** (prices can be expressed in pound and pennies)
- Include the same columns in the Savoury table. In addition to that add a column called **main_ingredient** (it will be a descriptive word)
- Let's do it together!

INSERT

- We now know how to create tables, but we also need to be able to populate them with data
- The actual command syntax is INSERT INTO, meaning that we are inserting values into a table
- The command is followed by a table name (only one table is allowed)
- Statement is used to add new rows of data to a table



SYNTAX

```
INSERT INTO table_name  
(col1, col2, col3)  
VALUES  
(val1, val2, val3);
```

EXAMPLE

```
INSERT INTO  
customers  
(name, surname, telephone)  
VALUES  
(‘Mary’, ‘Jones’, ‘123-4567’)
```

BULK INSERT

- INSERT allows only one table and column list
- Insert multiple rows with one statement
- Inserting multiple rows can be achieved by inserting multiple value lists
- Alternatively we can use SELECT statement following table name to perform bulk insert



SYNTAX

```
INSERT INTO table_name  
(col1, col2, col3)  
VALUES  
(val1, val2, val3),  
(val4, val5, val6),  
(val7, val8, val9);
```

EXAMPLE

```
INSERT INTO  
customers  
(name, surname, telephone)  
VALUES  
(‘Mary’, ‘Jones’, ‘123-4567’),  
(‘Julie’, ‘Smith’, ‘123-0000’)  
(‘Joanna’, ‘Bates’, ‘777-8888’);
```

□ INSERT INTO
TABLE

□ COLUMN NAMES
VALUES KEYWORD

□ MULTIPLE VALUES LIST (COMMA
SEPARATED)



SYNTAX

We can use a select statement to retrieve data from one table and paste it into another as long as their schemas match.

```
INSERT INTO table_name  
(col1, col2, col3)  
  SELECT *  
  FROM old_table_name  
  WHERE  
old_table_name.col > 10;
```

EXAMPLE

```
INSERT INTO  
customers  
SELECT *  
FROM old_customers  
WHERE  
old_customers.cust_id > 10;
```

- ☐ INSERT INTO
- ☐ TABLE
- ☐ SELECT STATEMENT

PRACTICE



We need to populate tables that we created in the Bakery database. At the moment they are like empty shells. Let's add some data to get the result below.

TASKS

Make Sweet table look like this

id	item_name	price
1	doughnut	0.5
2	croissant	0.75
3	painauchocolat	0.55
4	cinnamon swirl	0.45
5	cannoli	0.88
6	apple tart	1.12

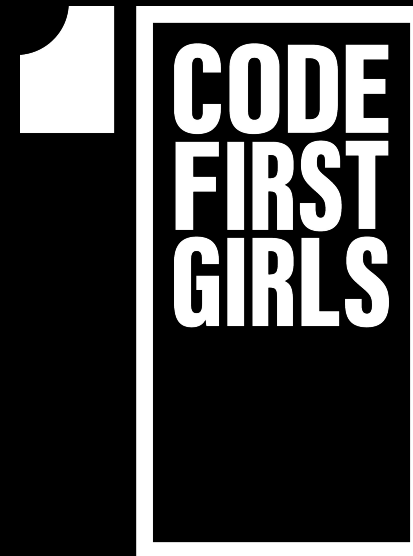
Make Savoury table look like this

id	item_name	price	main_ingredient
1	meat pie	1.25	pork
2	sausage roll	1	NULL
3	pasty	2.45	beef

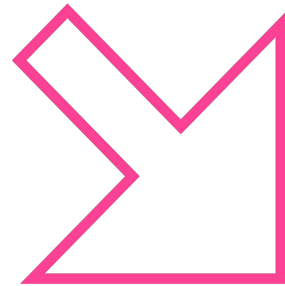
INDEPENDENT PRACTICE



- Revise the slides and practice to write SELECT queries using default databases or the database that we created in class



THANK YOU!




REFERENCE MATERIALS



WHY SQL & DATABASES




- SQL is one of the most sought-after skills by hiring employers in the tech world!
- It is a **lucrative skill by itself** or in combination with other development or analytical skills
- SQL developers and data scientists are in high demand!
- You must know how to write SQL queries **if you want to be a backend or frontend developer**
- SQL is a standard language for working with databases (it has been the industry standard for the past 30-40 years!)
- You can access and work with more data faster
- **SQL is easy** and it is also the first step towards data science career!




Data Analyst

DB and data are core and a must to know.




Data Scientist

Must understand, retrieve and use data better than anyone!




Data Engineer

Architects and designs databases, optimises data engines.



DBA


Essential skill to maintain and manage databases, as well as grant access.



Backend Developer

Backend ALWAYS works with data. Must have skills similar to Data Engineer

WHO SHOULD LEARN SQL AND DATABASES?




Frontend Developer

Frontend frameworks
Angular and React use direct interaction with databases



Mobile App Developer

Uses data storage techniques. Must know about SQL and databases



Product Manager

Need to know inside out the products they own. Data speaks truth about product's health.



Marketing Specialist

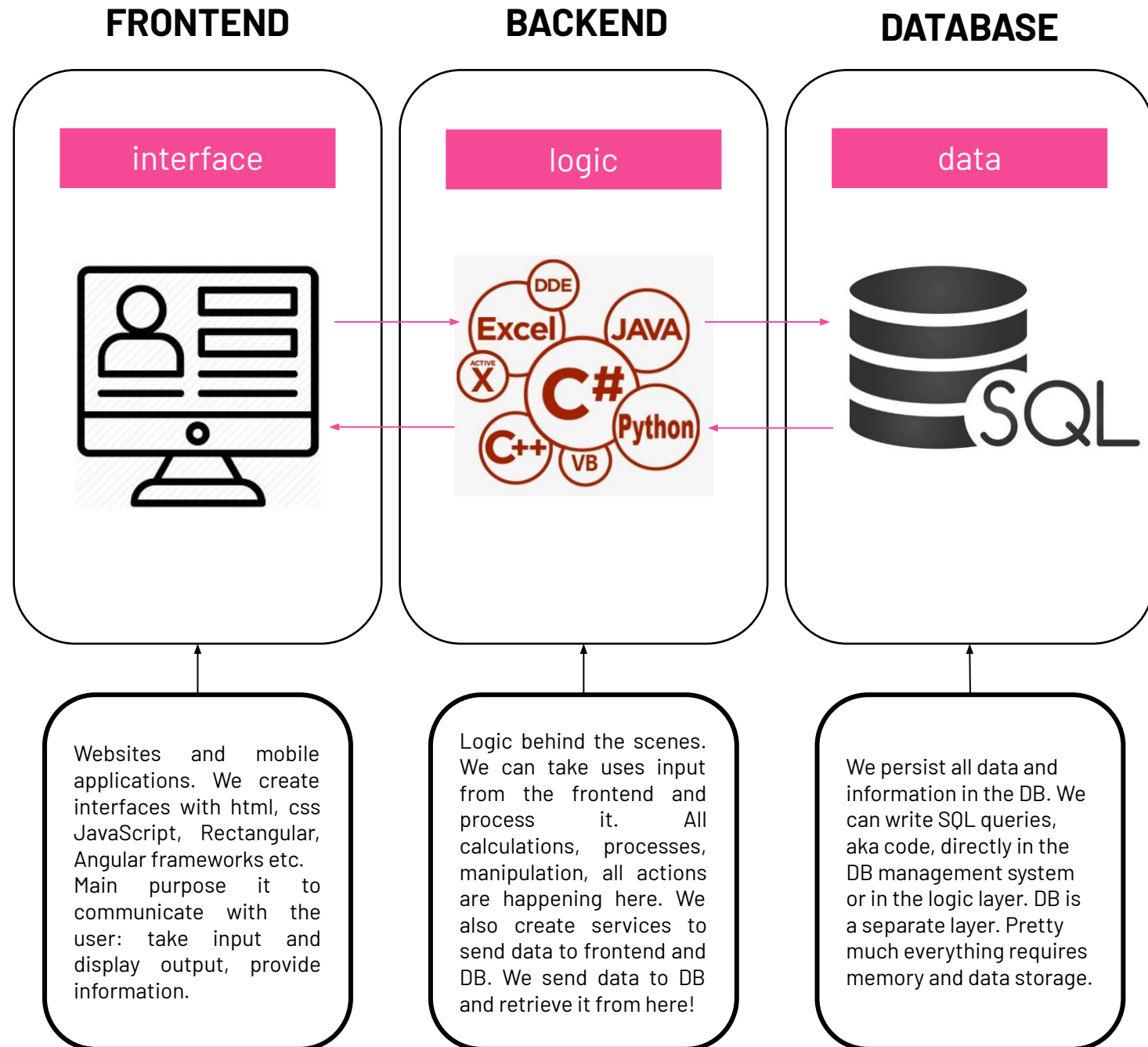
Data-driven profession.
Self-serve by retrieving and analysing data



Business Analyst

Advise on and make business decisions based on data analysis.

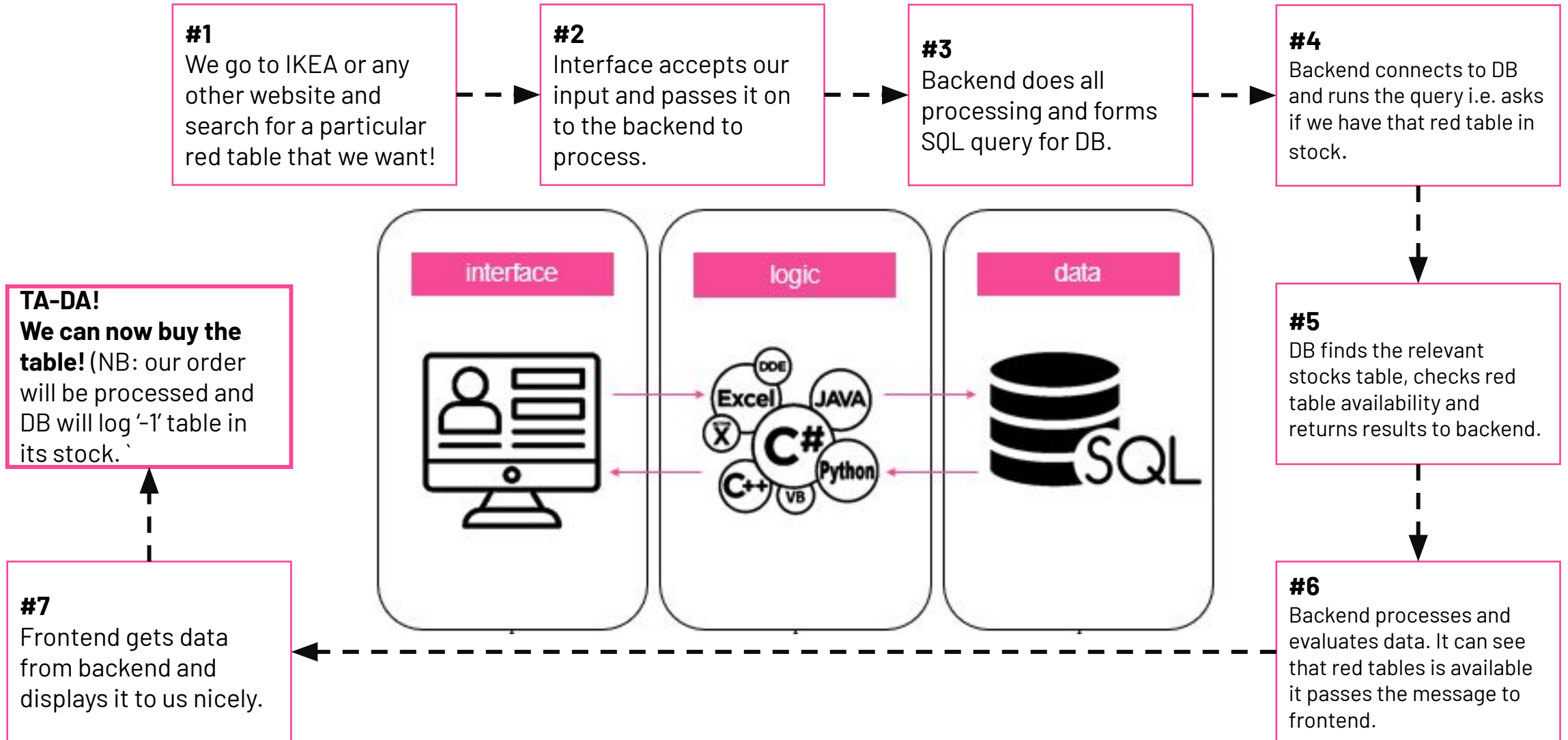
LAYERS MODEL





EXAMPLE 1 – LOCAL DB

Let's order some new furniture, maybe a table from an online shop!





EXAMPLE 1 – REMOTE DB WITH API

Let's imagine that we are going on holidays and want to book flights to sunny Costa Rica.



#1

We go to KAYAK or Skyscanner website and input our flight search criteria.

#2

Interface accepts our input and passes it on to the backend to process.

#3

Backend does all processing, setup filters and forms a request for API to retrieve data.

#4

A call to API is made to request data about flights from databases.

#5

API calls database, gets data and returns a bulk of data to backend.

#6

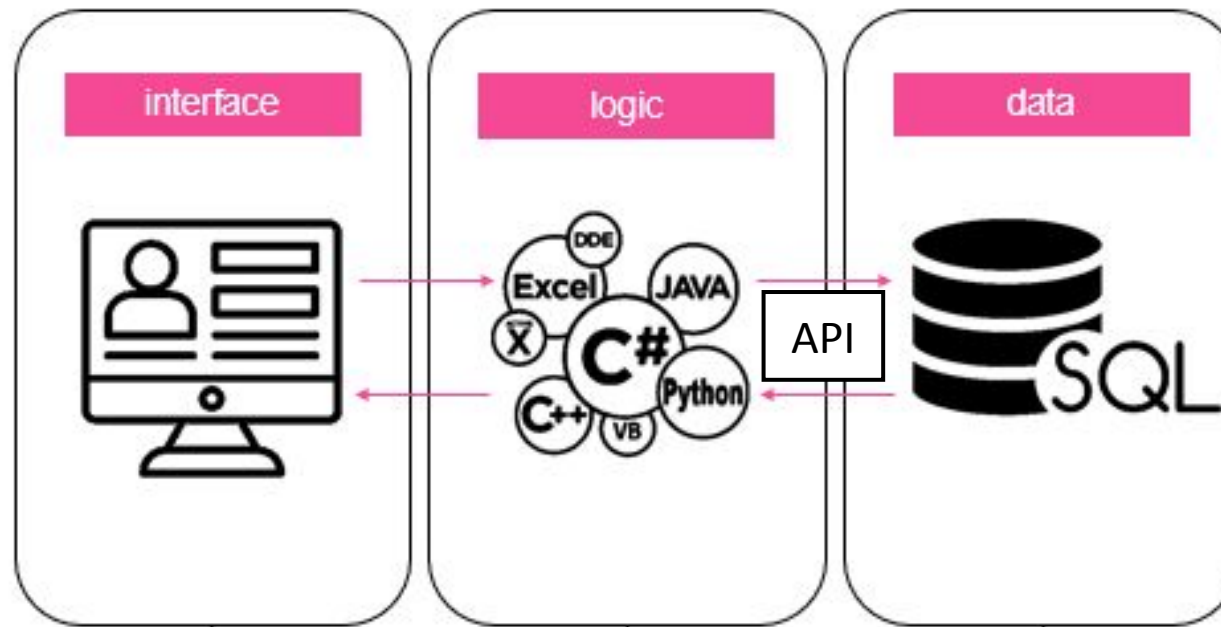
Backend processes data, forms it into logical output and passes it back to frontend.

TA-DA!

We choose a flight to go and see sloths 😊.

#7

Frontend gets data from backend and displays it to us nicely.





Structured Query Language (SQL) is a special-purpose programming language

SPECIAL PURPOSE



- SQL is used to communicate with a database
- It manipulates and manages sets of data
- Relational Database contains sets of data
- SQL statements (code) are used to perform tasks on the datasets in a database

DATABASE (SHORT DB)



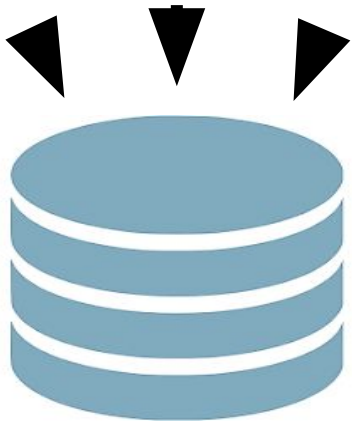
- Container that holds data
- Data in the DB container is organised in a particular way
- Data is held in tables (each table has columns and rows, similar to a spreadsheet). One database holds many tables
- Data can be *efficiently* stored and retrieved from DB

DATABASE VS SPREADSHEET



- Commonality: data stored in table as rows and columns
- DB can contain hundreds of tables and all those tables can be related to one another
- DB can hold data for the whole line of business, e.g. HR, Finance, Sales
- We can easily retrieve data from a DB, including data from multiple tables combined at the same time. We can also update and insert data more efficiently

DATA IN DATABASE



- We want to put data in a DB in such way that it makes the one and only source of truth
- We want data to be accessible and we want it to be easy to 'query', i.e. ask the DB questions about our data
- We want to add new data and delete old, unused one, so we want to manage and maintain our data efficiently
- We want to persist data and keep it safe

QUICK SUMMARY



- Databases are one of the most important parts of the tech world. It is widely used in programming, software development, application development and many other areas
- We use SQL language to communicate with Databases, specifically, we ask DB questions to retrieve data
- Relational Database holds many tables that have connections to each other, i.e. tables are related to one another in a certain way
- Data is stored in tables. Each table has columns, rows, cells and restrictions applied to them
- There are many different types of data that can be stored in table columns. There are also NULL values, which mean that 'in theory data exist, but it is not available in the DB'