



Data Analytics

Agenda

1. IBM Coursera Course - W1
2. Standard Deviation
3. Accessing Databases with Python
4. Business Intelligence (BI)



IBM Coursera Course - W1

Importing Data

- Process of loading and reading data into Python from various resources.
- **Two important properties:**
 - Format
 - various formats: .csv, .json, .xlsx, .hdf
 - File Path of dataset
 - Computer: /Desktop/mydata.csv
 - Internet: <https://archive.ics.uci.edu/autos/imports-85.data>

Importing a CSV into Python

```
import pandas as pd
```

```
url = "https://archive.ics.uci.edu/ml/machine-learningdatabases autos/imports-85.data"
```

```
df = pd.read_csv(url)
```


Importing a CSV into Python without a Header

```
import pandas as pd

url = "https://archive.ics.uci.edu/ml/machine-learning-databases/autos/imports-85.data"

df = pd.read_csv(url, header = None)
```

Printing the Dataframe in Python

- `df` prints the entire dataframe (not recommended for large datasets)
- `df.head(n)` to show the first n rows of data frame.
- `df.tail(n)` shows the bottom n rows of data frame.

```
df.head()
```

	0	1	2	3	4	5	6	7	8	9	...	16	17	18	19	20	21	22	23	24	25
0	3	?	alfa-romero	gas	std	two	convertible	rwd	front	88.6	...	130	mpfi	3.47	2.68	9.0	111	5000	21	27	13495
1	3	?	alfa-romero	gas	std	two	convertible	rwd	front	88.6	...	130	mpfi	3.47	2.68	9.0	111	5000	21	27	16500
2	1	?	alfa-romero	gas	std	two	hatchback	rwd	front	94.5	...	152	mpfi	2.68	3.47	9.0	154	5000	19	26	16500
3	2	164	audi	gas	std	four	sedan	fwd	front	99.8	...	109	mpfi	3.19	3.40	10.0	102	5500	24	30	13950
4	2	164	audi	gas	std	four	sedan	4wd	front	99.4	...	136	mpfi	3.19	3.40	8.0	115	5500	18	22	17450

Adding Headers - 1

- Replace default header (by `df.columns = headers`)

```
headers = ["symboling", "normalized-losses", "make", "fuel-type", "aspiration", "num-of-doors", "body-style",  
"drive-wheels", "engine-location", "wheel-base", "length", "width", "height", "curb-weight", "engine-type",  
"num-of-cylinders", "engine-size", "fuel-system", "bore", "stroke", "compression-ratio", "horsepower", "peak-  
rpm", "city-mpg", "highway-mpg", "price"]
```

```
df.columns=headers
```

```
df.head(5)
```


Adding Headers - 2

- Replace default header (by `df.columns = headers`)

	symboling	normalized-losses	make	fuel-type	aspiration	num-of-doors	body-style	drive-wheels	engine-location	wheel-base	...	engine-size	fuel-system	bore	stroke	compression-ratio	horsepower
0	3	?	alfa-romero	gas	std	two	convertible	rwd	front	88.6	...	130	mpfi	3.47	2.68	9.0	113
1	3	?	alfa-romero	gas	std	two	convertible	rwd	front	88.6	...	130	mpfi	3.47	2.68	9.0	113
2	1	?	alfa-romero	gas	std	two	hatchback	rwd	front	94.5	...	152	mpfi	2.68	3.47	9.0	113
3	2	164	audi	gas	std	four	sedan	fwd	front	99.8	...	109	mpfi	3.19	3.40	10.0	102
4	2	164	audi	gas	std	four	sedan	4wd	front	99.4	...	136	mpfi	3.19	3.40	8.0	115

Exporting a Dataframe to CSV

- Preserve progress anytime by saving modified dataset using

```
path="C:/Windows/.../ automobile.csv"
```

```
df.to_csv(path)
```


Exporting to Different Formats in Python

Data Format	Read	Save
csv	<code>pd.read_csv()</code>	<code>df.to_csv()</code>
json	<code>pd.read_json()</code>	<code>df.to_json()</code>
Excel	<code>pd.read_excel()</code>	<code>df.to_excel()</code>
sql	<code>pd.read_sql()</code>	<code>df.to_sql()</code>

Basic Insights from the Data

- Understand your data before you begin any analysis
- Should check:
 - Data Types
 - Data Distribution
- Locate potential issues with the data

Basic Insights of Dataset - Data Types - 1

Pandas Type	Native Python Type	Description
object	string	numbers and strings
int64	int	Numeric characters
float64	float	Numeric characters with decimals
datetime64, timedelta[ns]	N/A (but see the datetime module in Python's standard library)	time data.

Why check data types?

- potential info and type mismatch
- compatibility with python methods

Basic Insights of Dataset - Data Types - 2

- In pandas, we use `dataframe.dtypes` to check data types

```
df.dtypes
```

symboling	int64
normalized-losses	object
make	object
fuel-type	object
aspiration	object
num-of-doors	object
body-style	object
drive-wheels	object
engine-location	object
wheel-base	float64
length	float64
width	float64
height	float64
curb-weight	int64
engine-type	object
num-of-cylinders	object
engine-size	int64
fuel-system	object
bore	object
stroke	object
compression-ratio	float64
horsepower	object
peak-rpm	object
city-mpg	int64
highway-mpg	int64
price	object
dtype:	object

dataframe.describe()

- Returns a statistical summary

```
df.describe()
```

	symboling	wheel-base	length	width	height	curb-weight	engine-size	compression-ratio	city-mpg	highway-mpg
count	205.000000	205.000000	205.000000	205.000000	205.000000	205.000000	205.000000	205.000000	205.000000	205.000000
mean	0.834146	98.756585	174.049268	65.907805	53.724878	2555.565854	126.907317	10.142537	25.219512	30.751220
std	1.245307	6.021776	12.337289	2.145204	2.443522	520.680204	41.642693	3.972040	6.542142	6.886443
min	-2.000000	86.600000	141.100000	60.300000	47.800000	1488.000000	61.000000	7.000000	13.000000	16.000000
25%	0.000000	94.500000	166.300000	64.100000	52.000000	2145.000000	97.000000	8.600000	19.000000	25.000000
50%	1.000000	97.000000	173.200000	65.500000	54.100000	2414.000000	120.000000	9.000000	24.000000	30.000000
75%	2.000000	102.400000	183.100000	66.900000	55.500000	2935.000000	141.000000	9.400000	30.000000	34.000000
max	3.000000	120.900000	208.100000	72.300000	59.800000	4066.000000	326.000000	23.000000	49.000000	54.000000

dataframe.describe(include="all")

- Provides full summary statistics
- ```
df.describe(include="all")
```

|        | symboling  | normalized-losses | make   | fuel-type | aspiration | num-of-doors | body-style | drive-wheels | engine-location | wheel-base | ... | engine-size | fuel-system | bore | stroke |
|--------|------------|-------------------|--------|-----------|------------|--------------|------------|--------------|-----------------|------------|-----|-------------|-------------|------|--------|
| count  | 205.000000 | 205               | 205    | 205       | 205        | 205          | 205        | 205          | 205             | 205.000000 | ... | 205.000000  | 205         | 205  | 205    |
| unique | NaN        | 52                | 22     | 2         | 2          | 3            | 5          | 3            | 2               | NaN        | ... | NaN         | 8           | 39   | 37     |
| top    | NaN        | ?                 | toyota | gas       | std        | four         | sedan      | fwd          | front           | NaN        | ... | NaN         | mpfi        | 3.62 | 3.40   |
| freq   | NaN        | 41                | 32     | 185       | 168        | 114          | 96         | 120          | 202             | NaN        | ... | NaN         | 94          | 23   | 20     |
| mean   | 0.834146   | NaN               | NaN    | NaN       | NaN        | NaN          | NaN        | NaN          | NaN             | 98.756585  | ... | 126.907317  | NaN         | NaN  | NaN    |
| std    | 1.245307   | NaN               | NaN    | NaN       | NaN        | NaN          | NaN        | NaN          | NaN             | 6.021776   | ... | 41.642693   | NaN         | NaN  | NaN    |
| min    | -2.000000  | NaN               | NaN    | NaN       | NaN        | NaN          | NaN        | NaN          | NaN             | 86.600000  | ... | 61.000000   | NaN         | NaN  | NaN    |
| 25%    | 0.000000   | NaN               | NaN    | NaN       | NaN        | NaN          | NaN        | NaN          | NaN             | 94.500000  | ... | 97.000000   | NaN         | NaN  | NaN    |
| 50%    | 1.000000   | NaN               | NaN    | NaN       | NaN        | NaN          | NaN        | NaN          | NaN             | 97.000000  | ... | 120.000000  | NaN         | NaN  | NaN    |
| 75%    | 2.000000   | NaN               | NaN    | NaN       | NaN        | NaN          | NaN        | NaN          | NaN             | 102.400000 | ... | 141.000000  | NaN         | NaN  | NaN    |
| max    | 3.000000   | NaN               | NaN    | NaN       | NaN        | NaN          | NaN        | NaN          | NaN             | 120.900000 | ... | 326.000000  | NaN         | NaN  | NaN    |

# Basic Insights of Dataset

`dataframe.info()` provides a concise summary of your DataFrame.

`df.info()`

Row Number

|     |     |     |            |        |       |
|-----|-----|-----|------------|--------|-------|
| 0   | 3   | 7   | alfa-romeo | gas    | std   |
| 1   | 3   | 7   | alfa-romeo | gas    | std   |
| 2   | 1   | 7   | alfa-romeo | gas    | std   |
| 3   | 2   | 164 | audi       | gas    | std   |
| 4   | 2   | 164 | audi       | gas    | std   |
| 5   | 2   | 7   | audi       | gas    | std   |
| 6   | 1   | 150 | audi       | gas    | std   |
| 7   | 1   | 7   | audi       | gas    | std   |
| 8   | 1   | 150 | audi       | gas    | turbo |
| 9   | 0   | 7   | audi       | gas    | turbo |
| 10  | 2   | 182 | bmw        | gas    | std   |
| 11  | 0   | 182 | bmw        | gas    | std   |
| 12  | 0   | 180 | bmw        | gas    | std   |
| 13  | 0   | 180 | bmw        | gas    | std   |
| 14  | 1   | 7   | bmw        | gas    | std   |
| 15  | 0   | 7   | bmw        | gas    | std   |
| 16  | 0   | 7   | bmw        | gas    | std   |
| 17  | 0   | 7   | bmw        | gas    | std   |
| 18  | 2   | 121 | chevrolet  | gas    | std   |
| 19  | 1   | 88  | chevrolet  | gas    | std   |
| 20  | 0   | 81  | chevrolet  | gas    | std   |
| 21  | 1   | 108 | dodge      | gas    | std   |
| 22  | 1   | 108 | dodge      | gas    | std   |
| 23  | 1   | 108 | dodge      | gas    | turbo |
| 24  | 1   | 148 | dodge      | gas    | std   |
| 25  | 1   | 148 | dodge      | gas    | std   |
| 26  | 1   | 148 | dodge      | gas    | std   |
| 27  | 1   | 148 | dodge      | gas    | turbo |
| 28  | -1  | 110 | dodge      | gas    | std   |
| 29  | 3   | 145 | dodge      | gas    | turbo |
| ... | ... | ... | ...        | ...    | ...   |
| 170 | -1  | 85  | toyota     | gas    | std   |
| 171 | -1  | 85  | toyota     | gas    | std   |
| 172 | -1  | 85  | toyota     | gas    | std   |
| 173 | 3   | 187 | toyota     | gas    | std   |
| 174 | 3   | 187 | toyota     | gas    | std   |
| 175 | -1  | 80  | toyota     | gas    | std   |
| 176 | -1  | 7   | toyota     | gas    | std   |
| 177 | 2   | 122 | volkswagen | diexel | std   |
| 178 | 2   | 122 | volkswagen | gas    | std   |
| 179 | 2   | 94  | volkswagen | diexel | std   |
| 180 | 2   | 94  | volkswagen | gas    | std   |
| 181 | 2   | 94  | volkswagen | gas    | std   |
| 182 | 2   | 94  | volkswagen | diexel | turbo |
| 183 | 2   | 94  | volkswagen | gas    | std   |
| 184 | 3   | 7   | volkswagen | gas    | std   |
| 185 | 3   | 256 | volkswagen | gas    | std   |
| 186 | 0   | 7   | volkswagen | gas    | std   |
| 187 | 0   | 7   | volkswagen | diexel | turbo |
| 188 | 0   | 7   | volkswagen | gas    | std   |
| 189 | -1  | 183 | volvo      | gas    | std   |
| 190 | -1  | 74  | volvo      | gas    | std   |
| 191 | -1  | 183 | volvo      | gas    | std   |
| 192 | -1  | 74  | volvo      | gas    | std   |
| 193 | -1  | 183 | volvo      | gas    | turbo |
| 194 | -1  | 74  | volvo      | gas    | turbo |
| 195 | -1  | 95  | volvo      | gas    | std   |
| 196 | -1  | 95  | volvo      | gas    | turbo |
| 197 | -1  | 95  | volvo      | gas    | std   |
| 198 | -1  | 95  | volvo      | diexel | turbo |
| 199 | -1  | 95  | volvo      | gas    | turbo |

This function shows the top 30 rows and bottom 30 rows of the data frame.

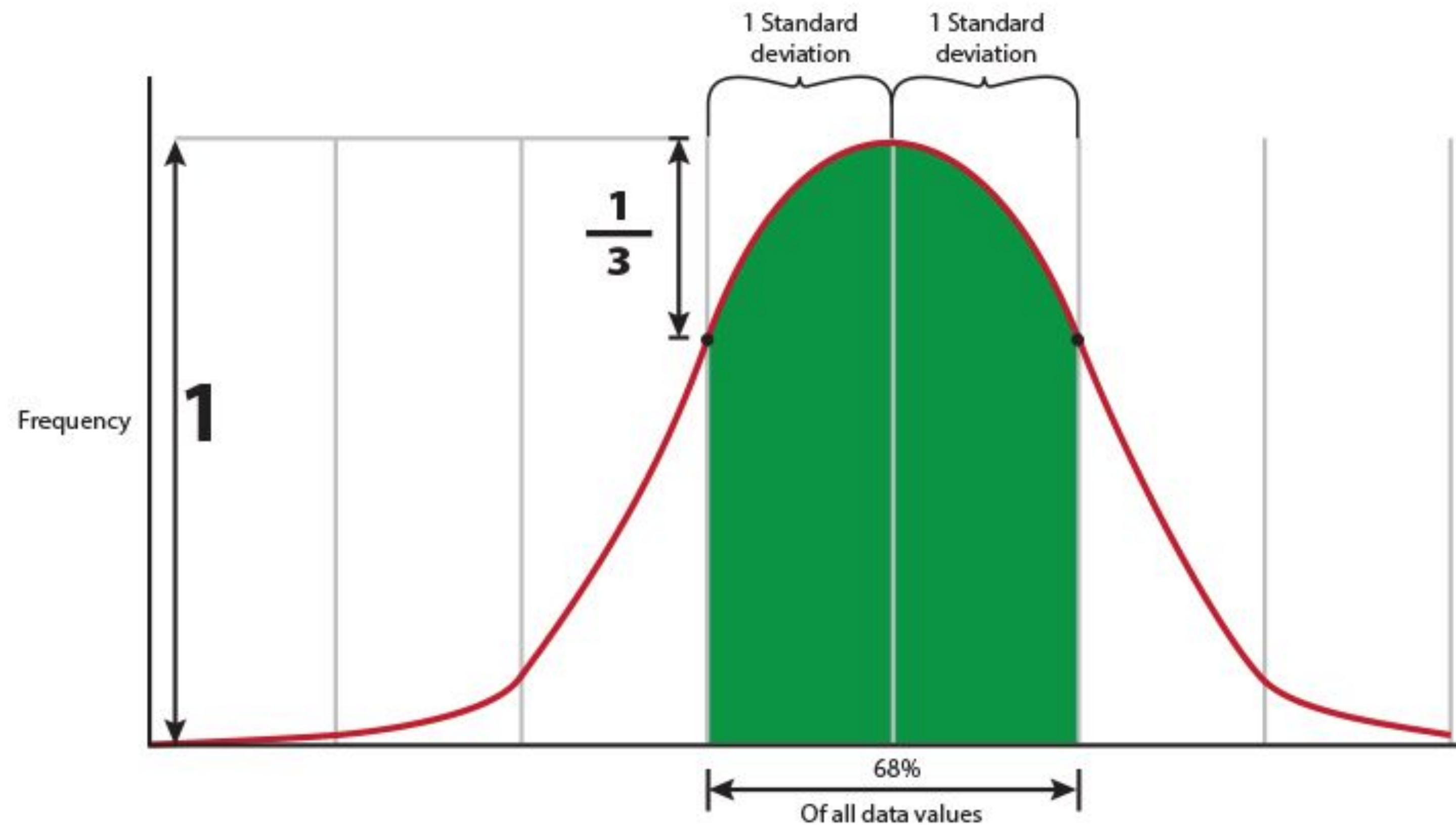




# Standard Deviation

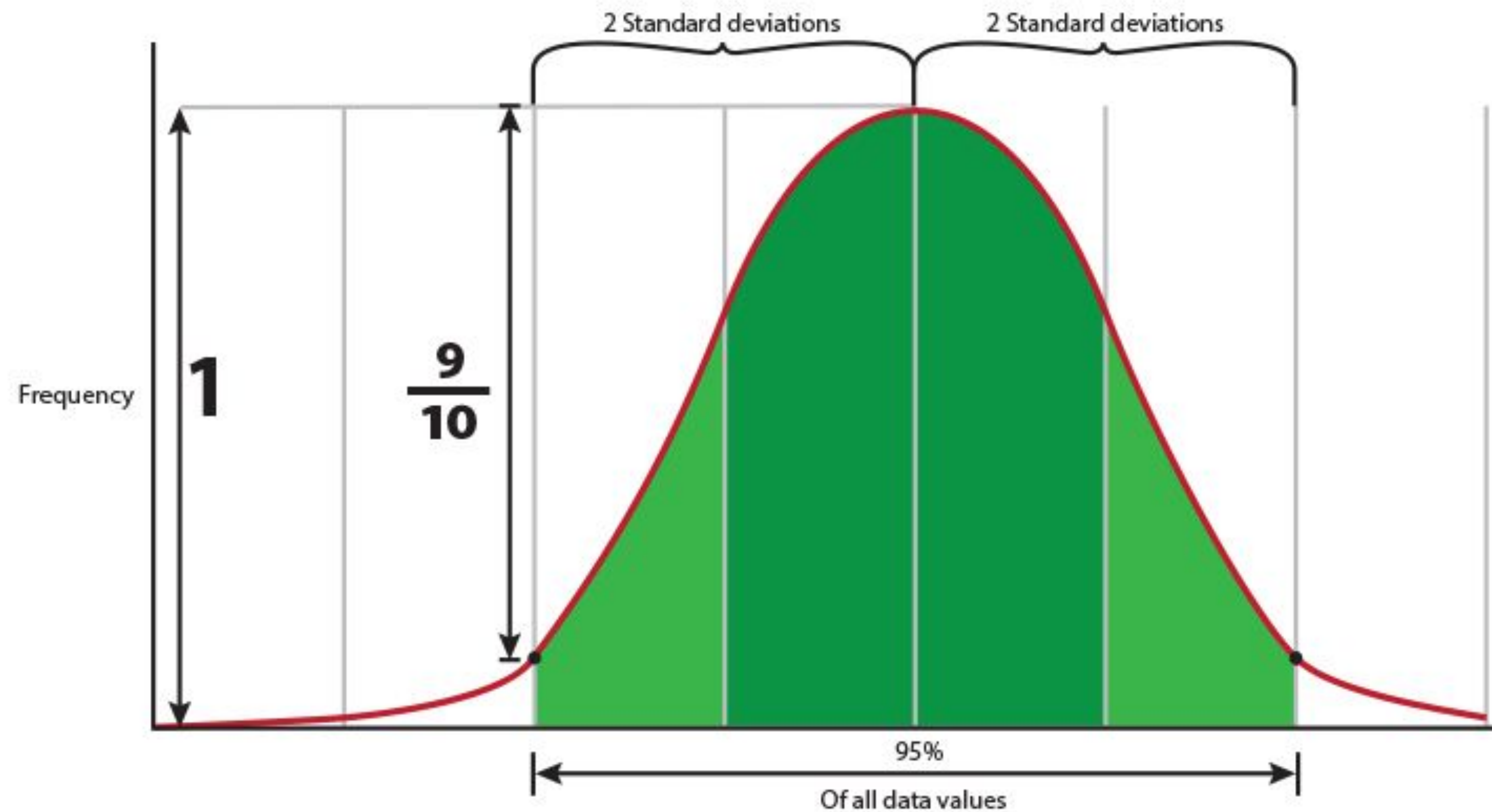
# Normal Distributions - Standard Deviation 1

**68%** of data is  $\leq 1$  standard deviation away from the mean



# Normal Distributions - Standard Deviation 2

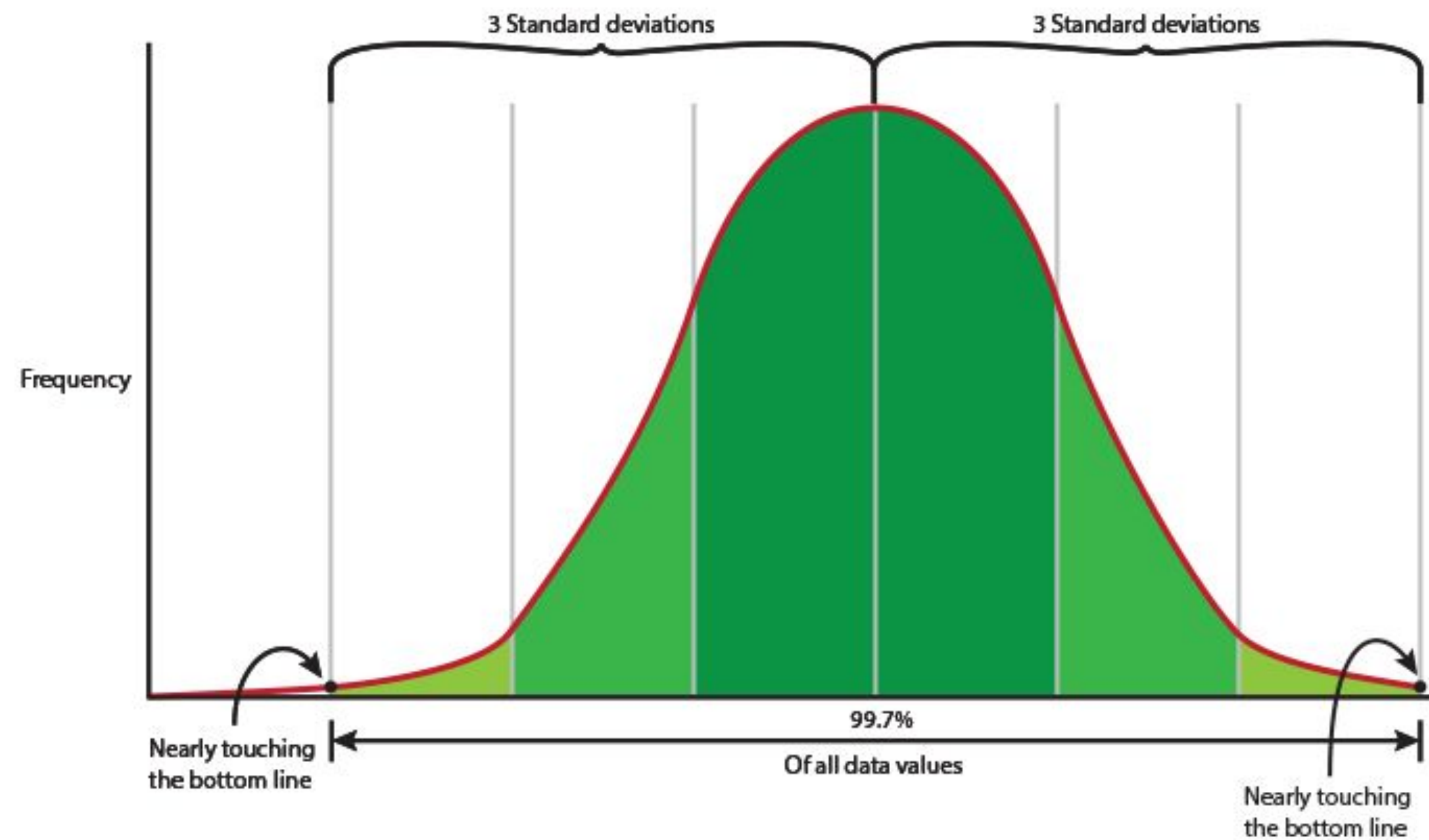
**95%** of data is  $\leq 2$  standard deviations away from the mean



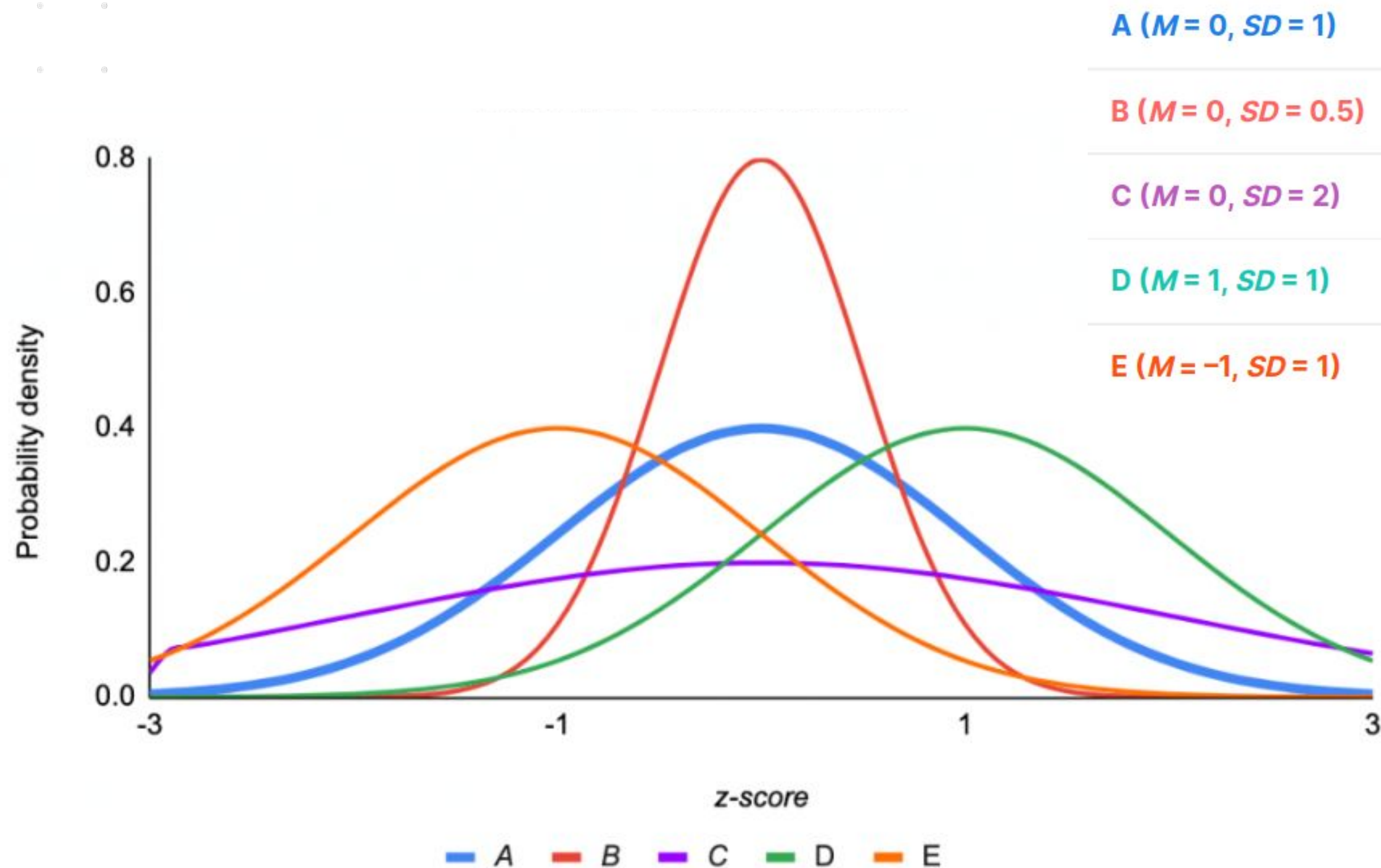


# Normal Distributions - Standard Deviation 3

**99.7%** of data is  $\leq 3$  standard deviations away from the mean



# Standard Normal Distribution

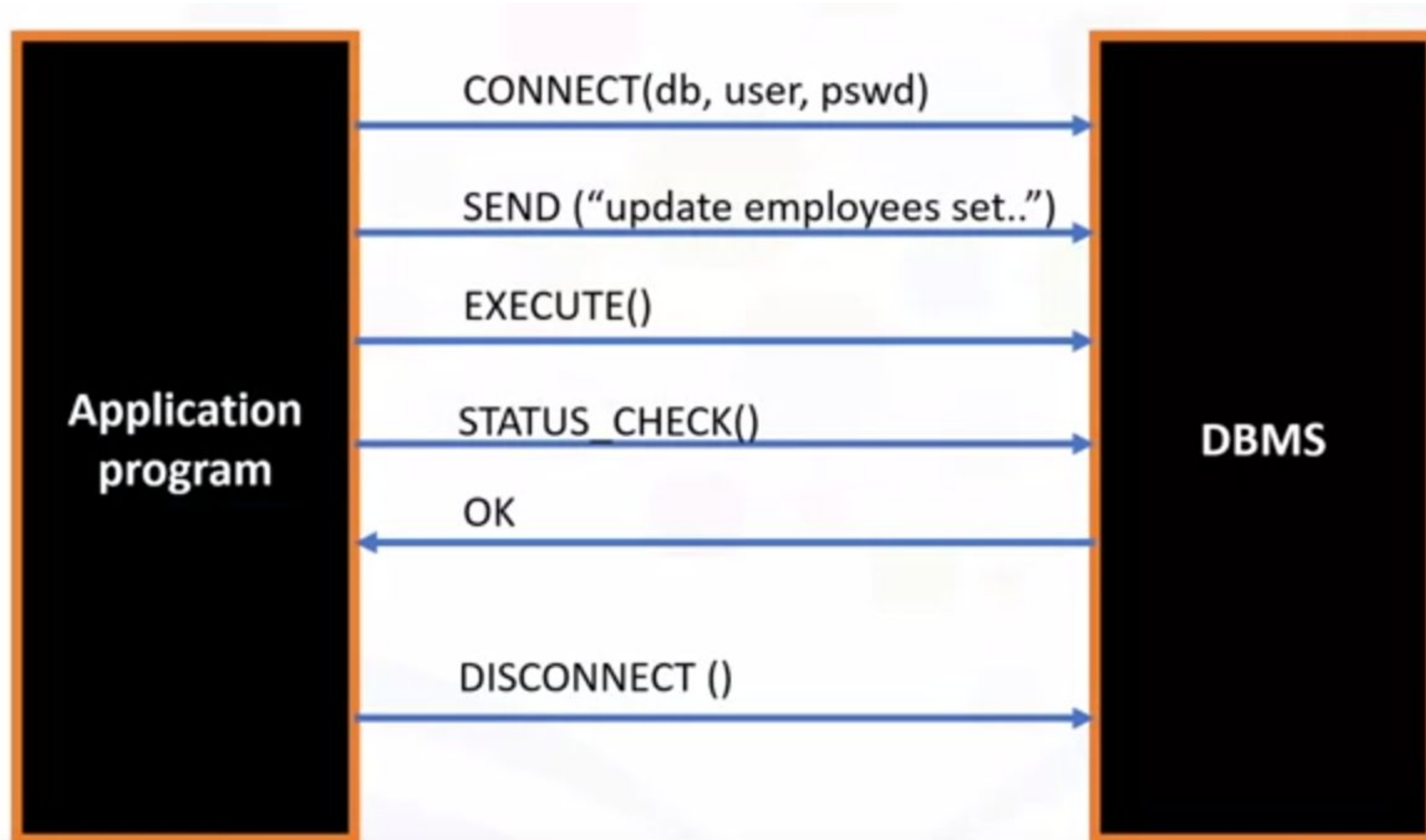




# Accessing Databases with Python

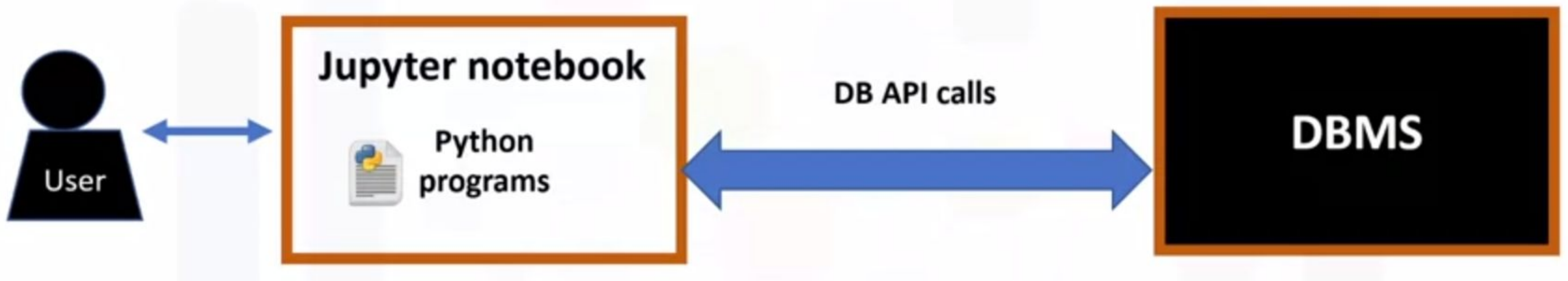


# SQL API



# Python DB API

- A standard that allows to write a single program
  - that works with multiple kinds of relational databases
  - instead of writing a separate program for each one



# Python DB API - Main Concepts 1

## 1) Connection Objects

- used to connect to a database and manage transactions
- The methods used with connection objects:
  - **cursor() method** returns a new cursor object using the connection
  - **commit() method** commit any pending transaction to the database
  - **rollback() method** roll back the database to the start of any pending transaction
  - **close() method** close a database connection



# Python DB API - Main Concepts 2

## 2) Cursor Objects

- used to run queries
- used to scan through the results of a database
- works similar to a cursor in a text processing system
  - where you scroll down in your result set and get your data into the application

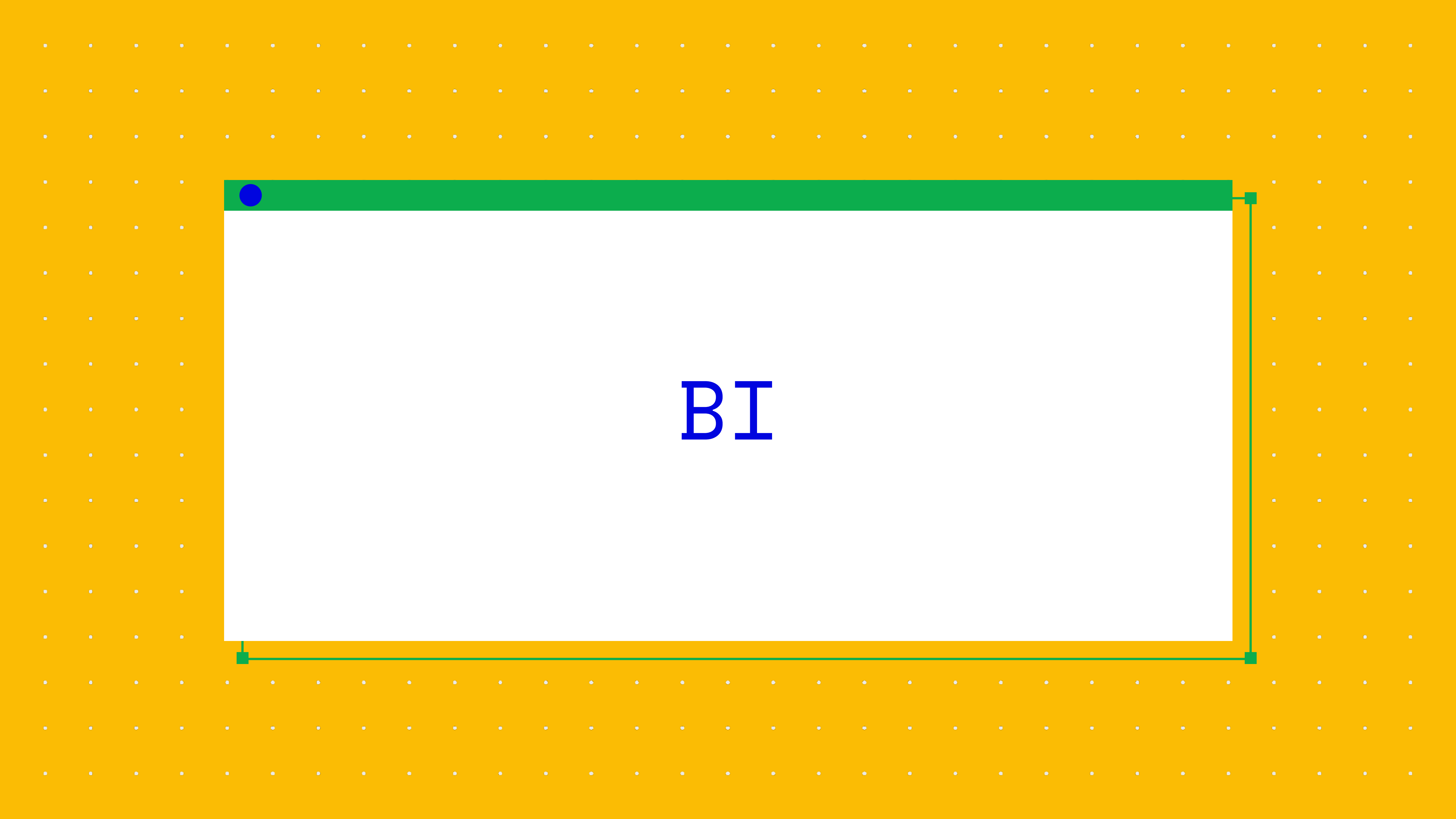
# Python DB API Example - MySQL

## MySQLdb

- is an interface for connecting to a MySQL database from Python
- implements the **Python DB API** v2.0

```
import MySQLdb

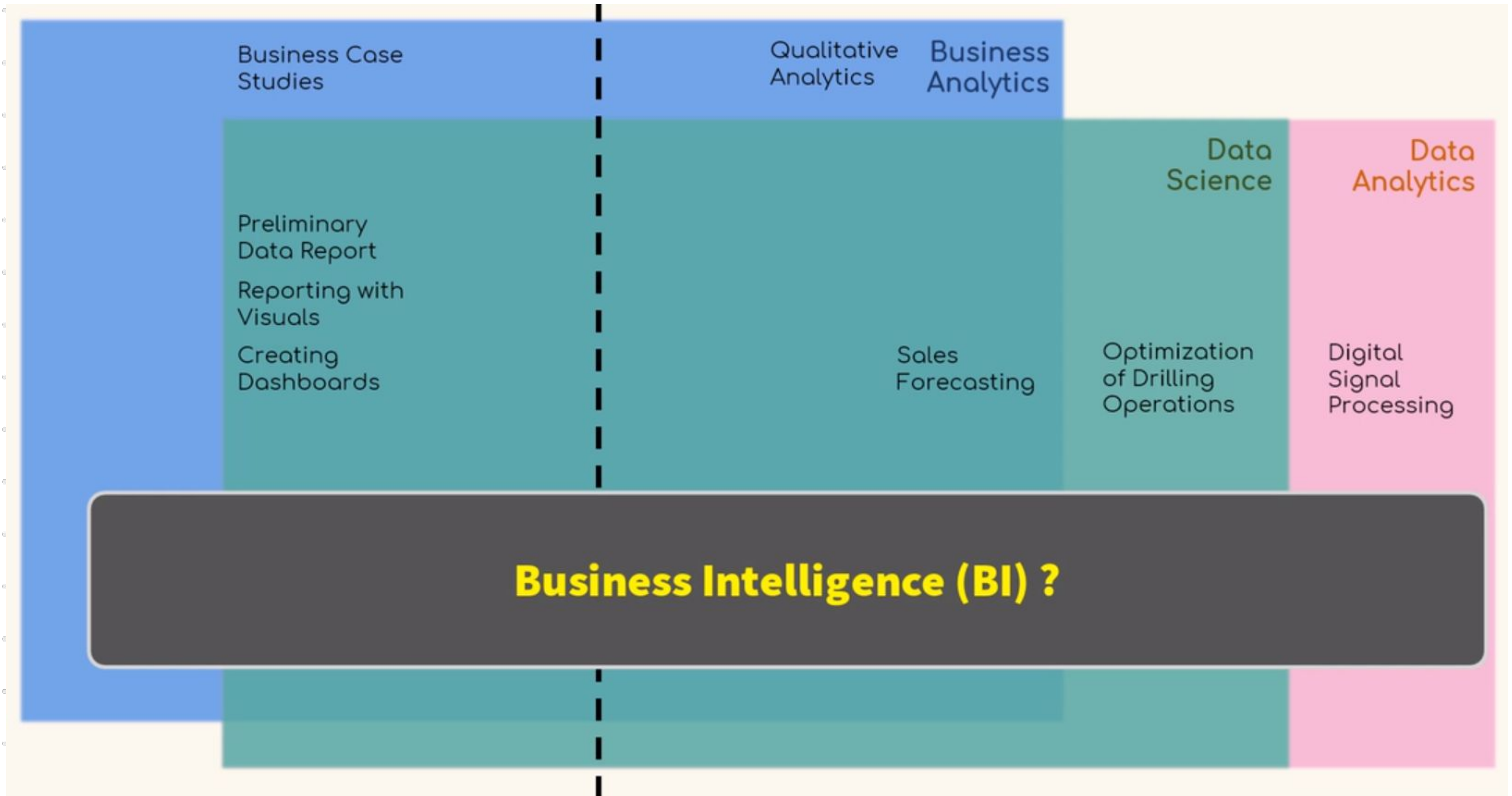
Open database connection
db = MySQLdb.connect("localhost", "testuser", "test123", "TESTDB")
```



BI



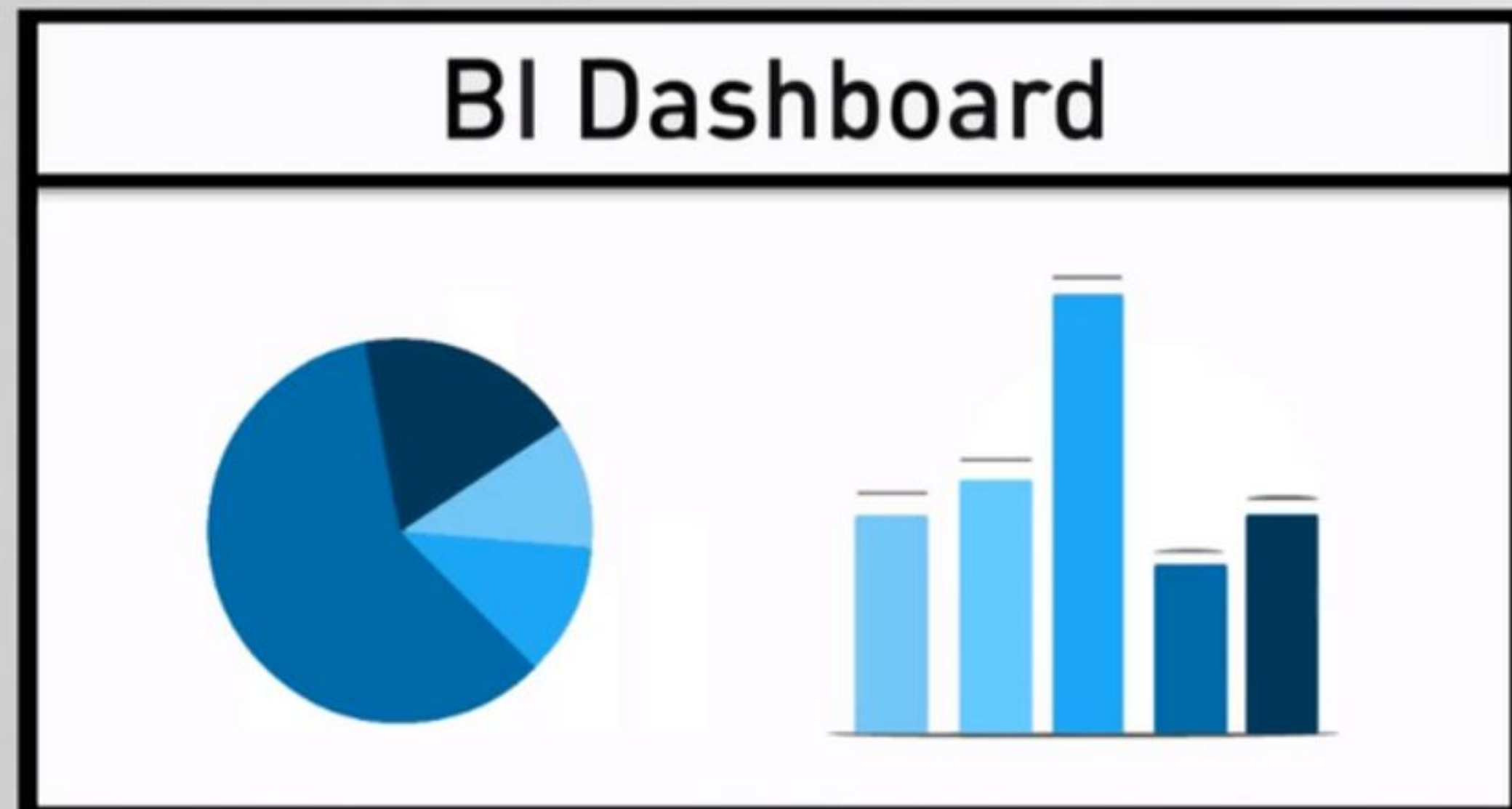
# Business Intelligence (BI) ?



# Business Intelligence (BI) - 1

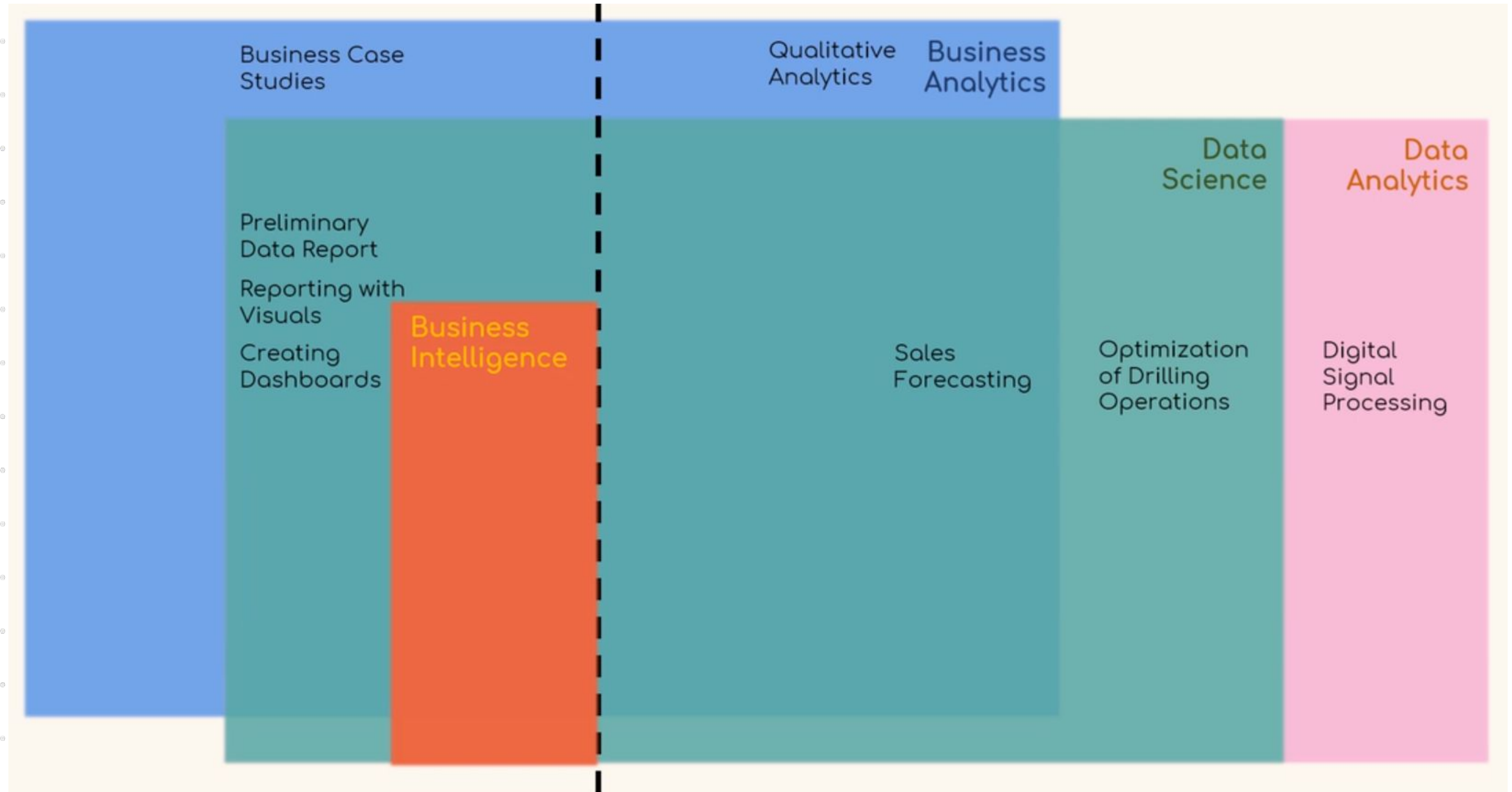
**business intelligence (BI):** the process of analysing and reporting historical business data

**aims to explain past events using business data**



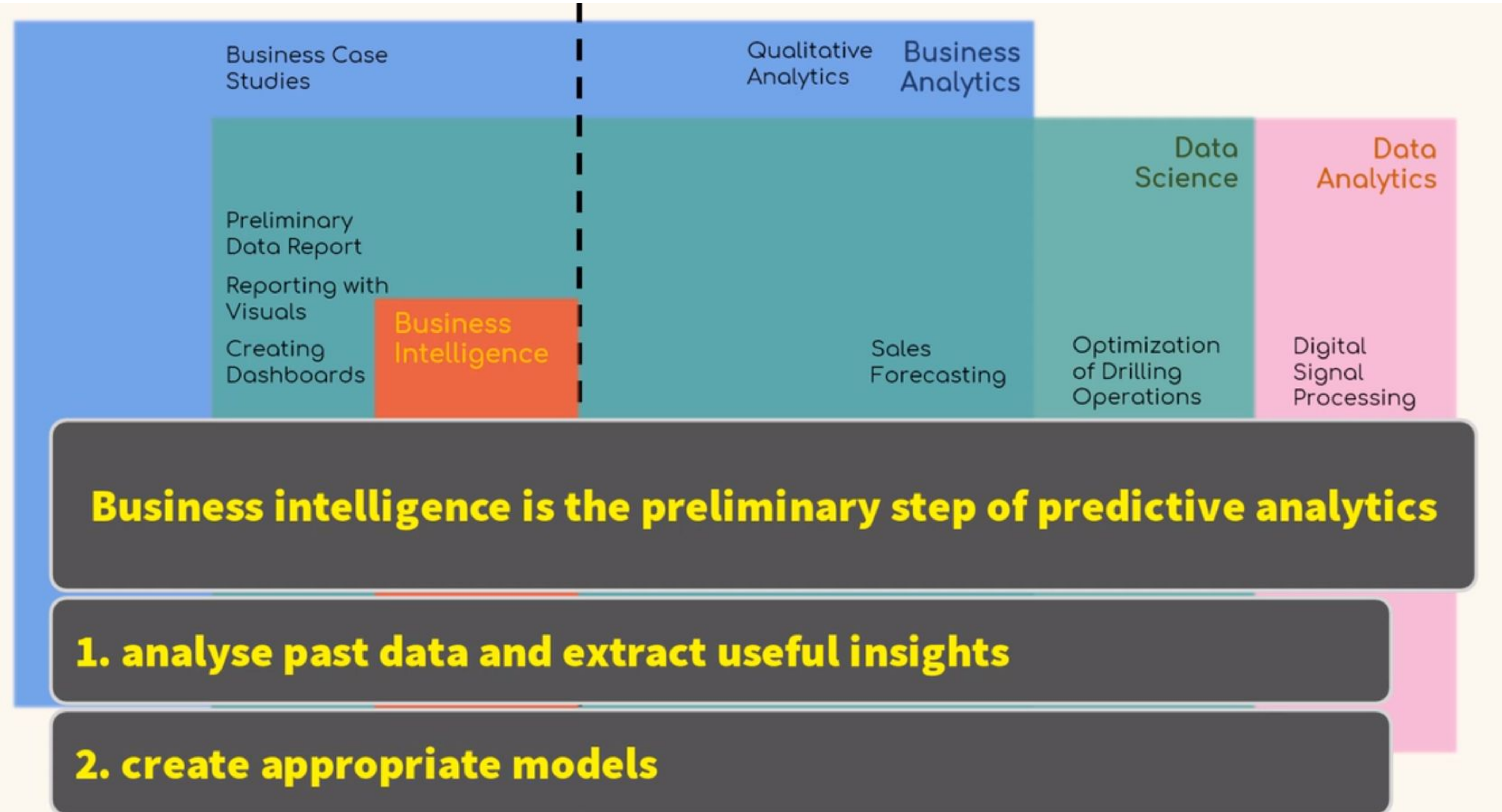


# Business Intelligence (BI) - 2





# Business Intelligence (BI) - 3



# Business Intelligence (BI) - 4

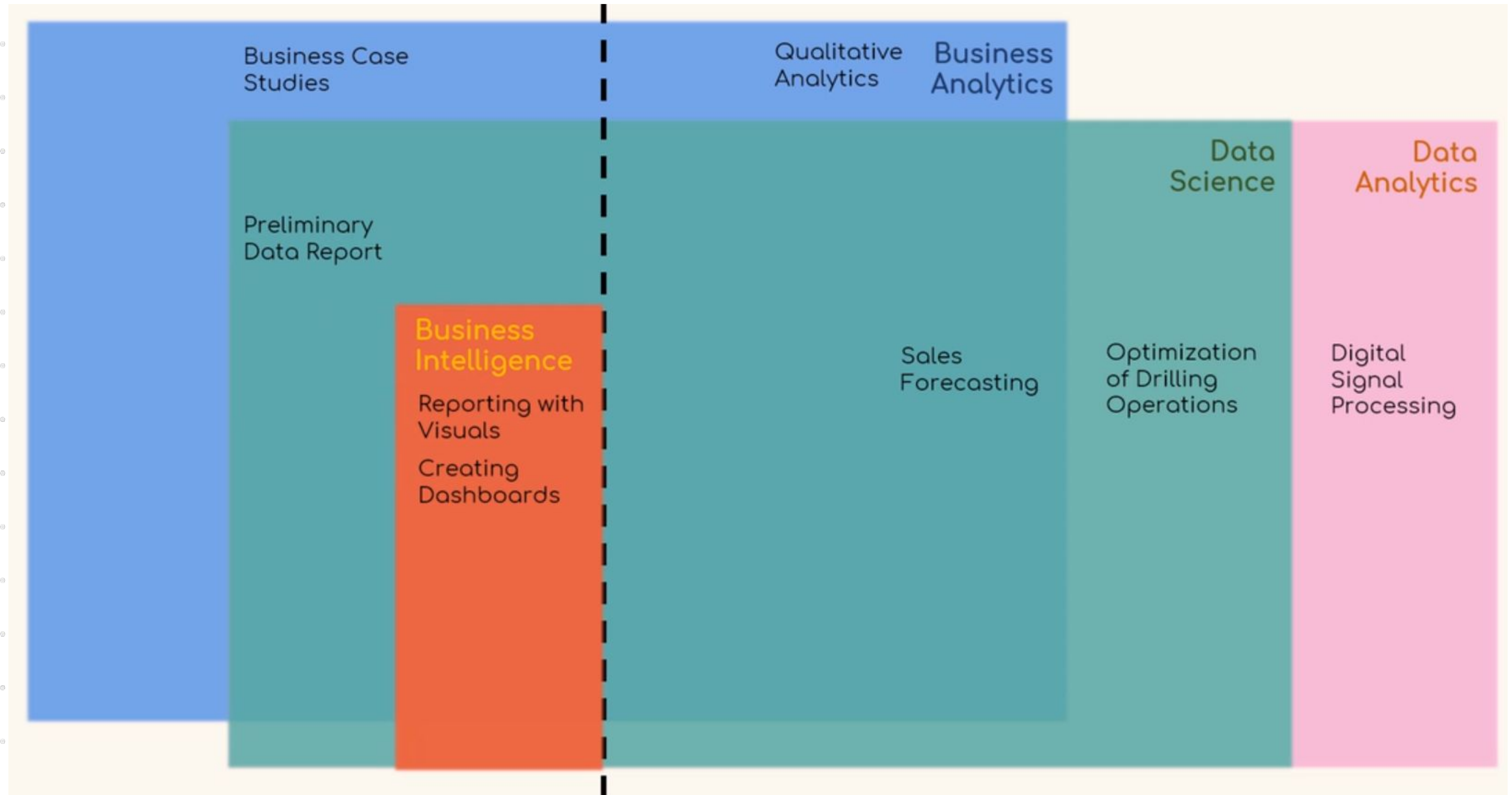
- analyse the **past** that you acquired (analyse past data)
- allows to make decisions, extract insights, and extract ideas
- includes all technology-driven tools involved in the process of:
  - analysing
  - understanding
  - reporting available past data

# Business Intelligence (BI) - 5

- After preparing Reports & Dashboards:
  - They can be used to make informed strategic & tactical business decisions
    - by end-users (such as general manager)
  - Using these inferences will allow to
    - create appropriate models that could predict the future of business accurately



# Business Intelligence (BI) - 6



# BI answers questions like

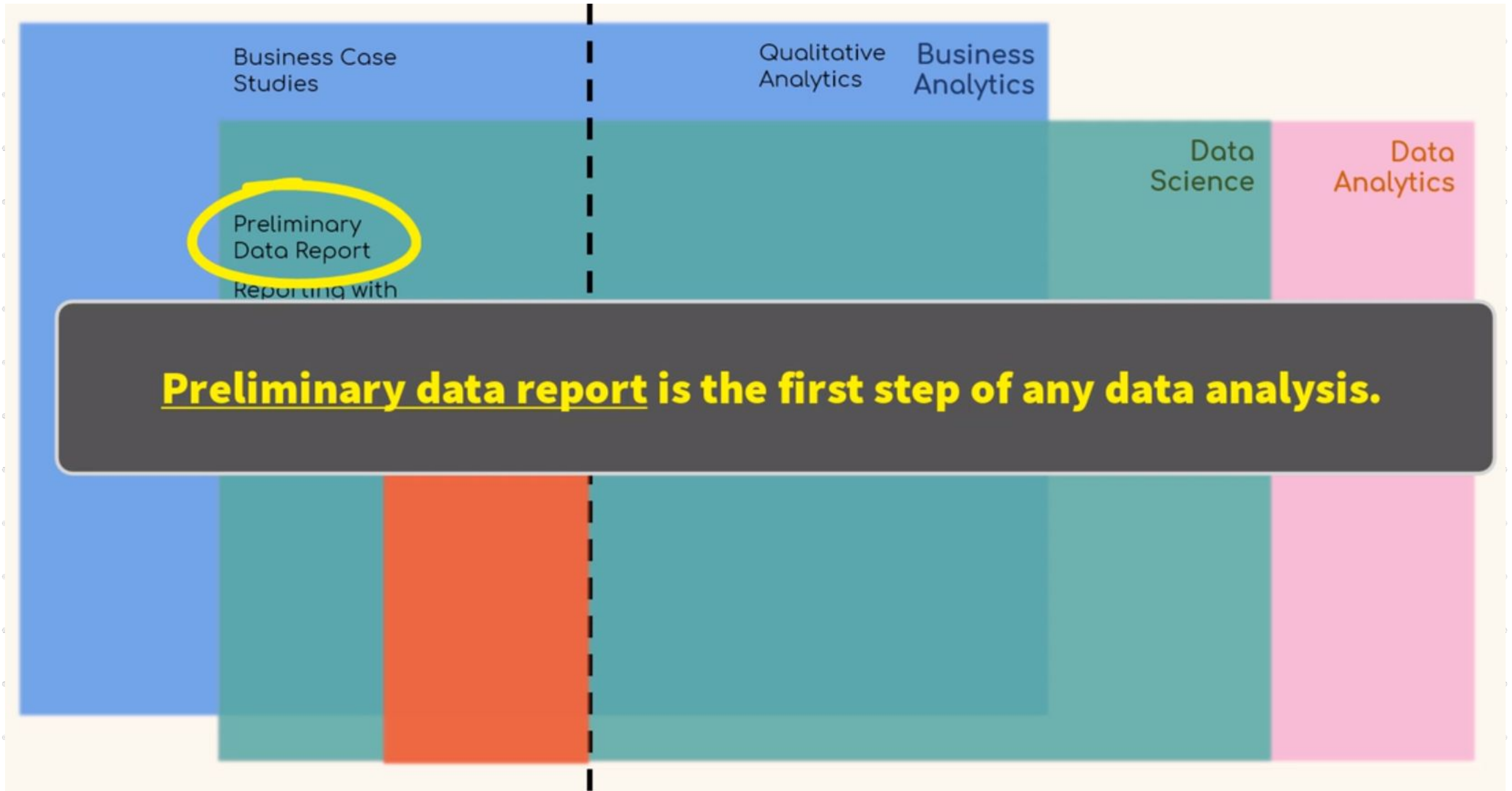
- What happened?
- When did it happen?
- How many units did we sell?
- In which region did we sell the most goods?

# BI Analyst

- **BI** requires the combination of
  - data skills
  - business knowledge
  - to explain the past performance of the company
- The job of a **BI Analyst** requires to:
  - understand the essence of a business
  - strengthen that business through the power of data




# Preliminary Data Report - 1



# Preliminary Data Report - 2

## Preliminary Data

- are data that do not have 12 months of reporting delay
- can apply to both annual and cumulative quarterly data



# Questions



# Links

<https://github.com/fcai-b/da>

# References

1. <https://learn.365datascience.com/courses/intro-to-data-and-data-science>
  - 365 Data Science - Introduction to Data and Data Science
2. <https://www.coursera.org/learn/data-analysis-with-python>
  - IBM Coursera Course - Data Analysis with Python