Applied Data Analytics

Pandas basics

Selecting rows with queries

Hans-Martin von Gaudecker and Aapo Stenhammar

Methods to select rows

- `.loc[]` / `.iloc[]` (yields individual items or slices)
  - Need to know the label(s) or location(s)
- `.loc[bool series]` / `[bool series]` (yields slices)
  - Quickly becomes cumbersome for complex conditions
  - Or need to construct the Boolean series upfront
- `query("condition")` (yields slices)
  - More flexible and often easier to read

Example: Gapminder data

```
import plotly.express as px

life_exp = px.data.gapminder()[["country", "year", "lifeExp"]]
```

	country	year	lifeExp
0	Afghanistan	1952	28.801
1	Afghanistan	1957	30.332
2	Afghanistan	1962	31.997
...	...	...	...
1701	Zimbabwe	1997	46.809
1702	Zimbabwe	2002	39.989
1703	Zimbabwe	2007	43.487

Simple query: Syntax

```
life_exp.query("country == 'Cuba'")
```

	country	year	lifeExp
384	Cuba	1952	59.421
385	Cuba	1957	62.325
386	Cuba	1962	65.246
...	...	...	...
393	Cuba	1997	76.151
394	Cuba	2002	77.158
395	Cuba	2007	78.273

Works for index or columns

```
life_exp = life_exp.set_index(["country", "year"])
life_exp.query("country == 'Cuba'")
```

country	year	lifeExp
Cuba	1952	59.421
Cuba	1957	62.325
Cuba	1962	65.246
...	...	...
Cuba	1997	76.151
Cuba	2002	77.158
Cuba	2007	78.273

Complex conditions are easy

```
life_exp.query("year > 2000 and country in ['Cuba', 'Spain']")
```

country	year	lifeExp
Cuba	2002	77.158
Cuba	2007	78.273
Spain	2002	79.78
Spain	2007	80.941

Using query results for setting values

Say we learn that life expectancy data for Spain after 2000 is unreliable and should be set to missing.

```
idx = life_exp.query("year > 2000 and country == 'Spain'").index
life_exp.loc[idx, "lifeExp"] = pd.NA
display(life_exp.query("country == 'Spain'").tail())
```

country	year	lifeExp
Spain	1987	76.9
Spain	1992	77.57
Spain	1997	78.77
Spain	2002	nan
Spain	2007	nan