Introduction to NoSQL

INTRODUCTION TO NOSQL

Jake Roach
AI Engineer

datacamp

Traditional relational data stores (RDBMS)

- Organize data in tables, using columns and rows
- Leverage SQL to manage and query data
- Enforce integrity through constraints on databases and tables

PostgreSQL

What is NoSQL?

Definition: NoSQL stands for "not only SQL", and is a set of data storage tools and techniques that allows for structured, semi-structured, and unstructured data to be stored and retrieved.

Characteristics:

- Allows for wide variety of data to be stored and retrieved
- Less rigid schema
- Better scaling and performance

NoSQL data stores

Tabular

title	pages	price
Data Analytics	528	28.60
R in Action	656	56.99
...	...	...
Practical SQL	464	27.99

- "Rectangular"
- Using columns and rows

Non-tabular

```
{
  "title": "Python for Data Analysis",
  "price": 53.99,
  ...
}
```

```
"weather": "sunny"
```

- Semi-structured format
- More flexible schema

NoSQL data stores

Column-oriented databases

A NoSQL data store that stores data by column, rather than row, and can be queried with SQL-like syntax. Allows for faster querying of data, especially when running analytical queries.

Use case: big data, analytics workflows

Document databases

NoSQL data storage tool used to store semi-structured "documents"

- JSON format
- Less rigid schema

Use case: user-generated data (reviews) and real-time analytics

More NoSQL data stores

Key-value

A NoSQL data storage tool that stores data as a collection of key-value pairs:

- Simple data that is written and read at a high frequency

Use cases: IoT (Internet of Things) data, mobile applications

Graph

A NoSQL data store that persists data in a network of nodes and edges.

- Nodes represent entities
- Edges represent relationships between entities

Use cases: social networks

Let's practice!

INTRODUCTION TO NOSQL

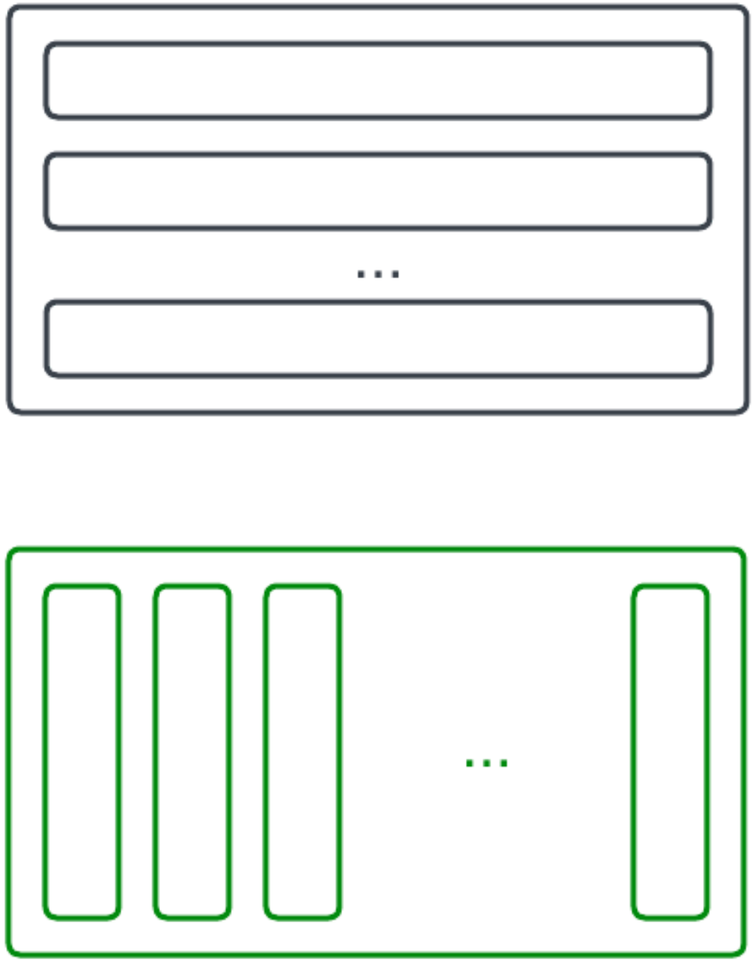
Tabular NoSQL data stores

INTRODUCTION TO NOSQL

Jake Roach
Data Engineer

datacamp

Tabular data stores



Column-oriented databases:

- Store data in columns, rather than rows
- Allow for selective column read and retrieval
- Easier schema changes
- Better data compression, query performance

Querying a column-oriented database

title	pages	price
Data Analytics	528	28.60
R in Action	656	56.99
...	...	...
Practical SQL	464	27.99

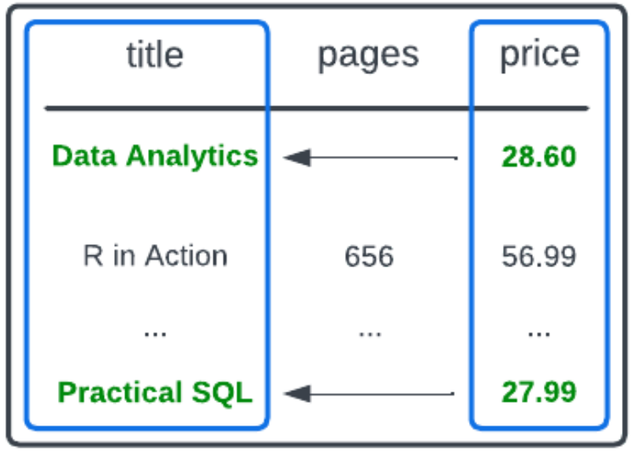
```
SELECT
    title,
    price
FROM books
WHERE price < 50.00;
```

title	pages	price
Data Analytics	528	**28.60**
R in Action	656	56.99
...	...	...
Practical SQL	464	**27.99**

- SQL-like syntax
- Column-elimination and selective reads/retrieval
- Automatic data clustering

Query execution in column-oriented data stores

```
SELECT
    title,
    price
FROM books
WHERE price < 50.00;
```



This query executes by:

- Accessing `price` column, identify records with `price < 50.00`
- Retrieving corresponding values from `title` column

Later, we'll look at:

- Optimizing data loads and deletes
- Creating performant `JOIN` s
- Working with semi-structure data

Connecting to a Snowflake database

```
import snowflake.connector


conn = snowflake.connector.connect(
  user="<user>",
  password="<password>",
  account="<account_identifier>",
  database="<database_name>",
  schema="<schema_name>",
  warehouse="<warehouse_name>"
)
```

- The `conn` variable will be created for you, pre-exercise

Writing and executing Snowflake queries

```
# Build a query in a string (or multi-line string)
query = """
SELECT
    title,
    price
FROM books
WHERE price < 50.00;
"""

# Execute the query, print the results
results = conn.cursor().execute(query).fetch_pandas_all()
print(results)
```

Let's practice!

INTRODUCTION TO NOSQL

Non-tabular NoSQL data stores

INTRODUCTION TO NOSQL

Jake Roach
Data Engineer

datacamp

Document databases

Definition: A NoSQL data storage tool that stores data in a flexible, semi-structured format, made up of key-value, key-array, and key-object pairs (similar to JSON).


```
{
    "title": "Python for Data Analysis",
    "price": 53.99,
    "topics": [
        "Data Science",
        "Data Analytics",
        ...
    ],
    "author": {
        "first": "William"
        ...
    }
}
```

Querying JSON data with Postgres JSON

books

```
{
    "title": "Python ...",
    "price": 53.99,
    "author": {
        "first": "William",
        "last": "McKinney"
    }
}
{
    "title": "Robust ...",
    "price": 32.99,
    "author": {
        "first": "Patrick",
        "last": "Viafore"
    }
}
```

```
SELECT
    books -> 'title' AS title,
    books -> 'price' AS price
FROM data_science_resources
WHERE
    books -> 'author' ->> 'last' = 'Viafore';
```

Resulting in the following output:

title	price
Robust Python ...	32.99

Connecting to a Postgres database

```
import sqlalchemy

# Create a connection string, and an engine
connection_string = "postgresql+psycopg2://<user>:<password>@<host>:<port>/<database>"
db_engine = sqlalchemy.create_engine(connection_string)
```

To create a connection to a Postgres database:

- Form a connection string
- Create an engine using `sqlalchemy.create_engine`
- `db_engine` variable will be created, pre-exercise

Writing and executing Postgres JSON queries

```
import pandas as pd

# Build the query
query = """
    SELECT
        books -> 'title' AS title,
        books -> 'price' AS price
    FROM data_science_resources;
"""

# Execute the query
result = pd.read_sql(query, db_engine)
print(result)
```

To write and execute a query:

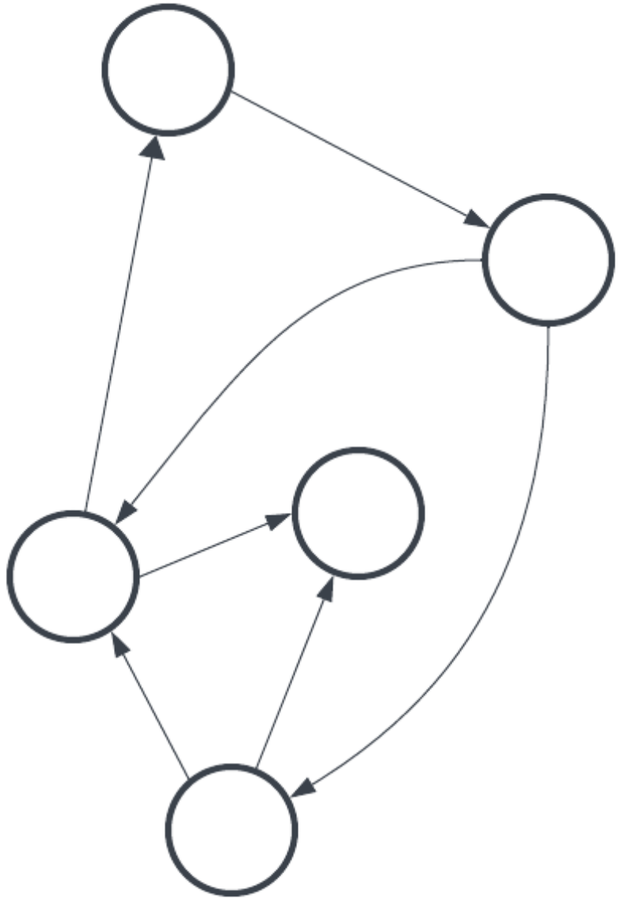
- Build a query in a string
- Pass `query` and `db_engine` to the `pd.read_sql()` function
- Print the resulting DataFrame

Other non-tabular NoSQL data stores

Key-value

```
{
    "name": "Jane Doe",
    "age": 25,
    "email": "jdoe@datacamp.com"
}
```

Graph



Let's practice!

INTRODUCTION TO NOSQL