Meerkat system training

Data Storage Solutions

Jyri Soppela

01/03/2017

Contents

1. Introduction

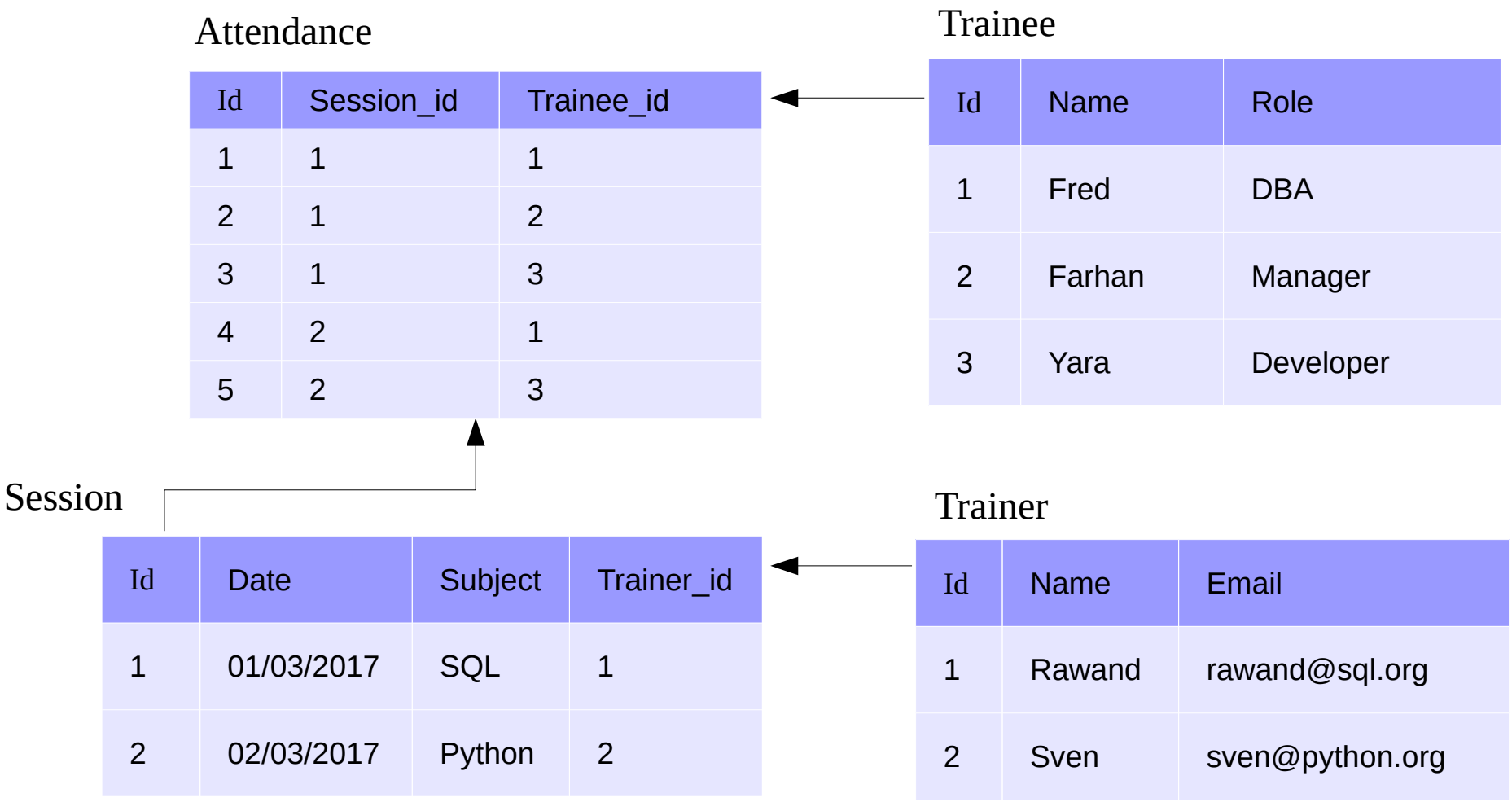
- Main aim of the Meerkat system is to provide fast public health surveillance
- There are several data sources:
  - ODK form data
  - Location and clinic data
  - Mobile device data
  - Variable data to calculate standardized variables based on raw data from ODK
- These data sources have slightly different data structure requirements
- The system uses several technologies for data storage and transfer:
  - Flat files
  - Relational Database: PostgreSQL
  - JSON databases: PostgreSQL and Amazon DynamoDB
-

2.1 Data types – Comma Separated Value

trainee_name	trainee_role	session_number	session_date	session_subject	trainer_name	trainer_email
Fred	DBA	1	01/03/2017	SQL	Rawand	rawand@sql.org
Farhan	Manager	1	01/03/2017	SQL	Rawand	rawand@sql.org
Yara	Developer	1	01/03/2017	SQL	Rawand	rawand@sql.org
Fred	DBA	2	02/03/2017	Python	Sven	sven@python.org
Yara	Developer	2	02/03/2017	Python	Sven	sven@python.org

- Flat files are easy to edit and display
- A good selection of software handles them: Excel, OpenOffice and LibreOffice all have spreadsheet capabilities and support the CSV format
- Flat files need to be validated to avoid conflicting data
- Relationship visualization is challenging

2.2 Data types – Relational database tables



2.3 Data types – Relational database tables

- Relational databases must be accessed via database clients
- Capacity is much higher than with flat files as the whole data set is not loaded to memory
- Relational table structure enforces the data format
- Data model maps out relationships and reduces conflicts in the data
- Relational databases are usually used via a language called Structured Query Language (SQL)
- A sample SQL data query:

```
SELECT NAME                    what data fields to get?
FROM TRAINEE                   where to get it from?
WHERE ROLE = 'DBA';            which rows to return?
```

2.4 Data types – JSON objects


```
[
  ▾ {
        session_number: 1,
        date: "2017-03-01",
        subject: "SQL",
      ▾ trainer: {
            name: "Rawand",
            email: "rawand@sql.org"
        },
      ▾ trainees: [
          ▾ {
                name: "Fred",
                role: "DBA"
            },
          ▾ {
                name: "Farhan",
                role: "Manager"
            },
          ▾ {
                name: "Yara",
                role: "Developer"
            }
        ]
    },
  ▾ {
        session_number: 2,
        date: "2017-03-02",
        subject: "Python",
      ▾ trainer: {
            name: "Sven",
            email: "sven@python.org"
        },
      ▾ trainees: [
          ▾ {
                name: "Fred",
                role: "DBA"
            },
          ▾ {
                name: "Yara",
                role: "Developer"
            }
        ]
    }
]
```


- JSON means JavaScript Object Notation even though it is used also outside JavaScript
- JSON objects consist of key-value pairs and ordered lists
- JSON objects can be nested
- Changing the structure of JSON objects does not require changes in the database structure
- Extremely high capacity databases use JSON structure but our usage is no where near this scale
- Mapping relationships is not as easy as with relational databases

Exercises: Building database queries with SQLAlchemy

- SQLAlchemy home page: HTTP://www.sqlalchemy.org
- SQLAlchemy reference manual: http://docs.sqlalchemy.org/en/rel_1_1/
- SQLAlchemy tutorials: http://www.sqlalchemy.org/library.html#tutorials
- SQLAlchemy JSON reference manual:

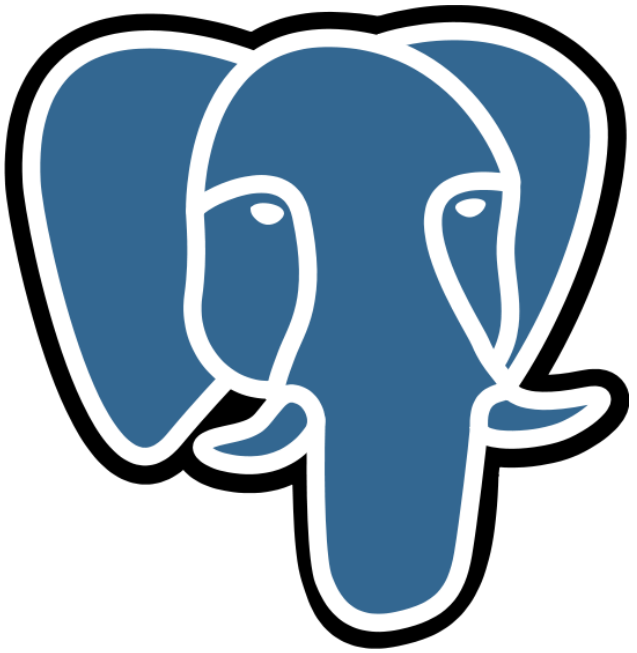
- Exercises are on Jupyter notebook in

  meerkat-code/training/data_storage_solutions/notebook

  - Since database connections require additional python packages, we run the notebook inside a Docker container
  - Text file docker-commands.sh has Docker commands for building and running the notebook container
  - Once the container is running, the notebook is available via browser at localhost:8889
  - Notebook name is SQLAlchemy

3 PostgreSQL

- PostgreSQL is an open source relational database
- In additional to standard SQL operations, PostgreSQL can store and query JSON objects
- JSON objects are stored in a separate JSON data type
- JSON operations are used as an extension to the SQL standard
- Other extensions of PostgreSQL include GIS location data and a text mining module



Exercise setup

- If you have a high bandwidth internet, take the following steps to build the docker image:
  - Go to training/data_storage_solutions/notebook
  - Run command
- Move the following ISO image file to your disk:
  - sql-ex.iso
- In Docker terminal, run the following command:
  - docker load –input sql-ex.iso
- If port 8888 has not been configured in virtual machine port configuration, do it now as per Meerkat setup instructions
- In Docker terminal, launch the exercise container with command:
  - docker run -it --rm -p 8888:8888 --network host sql-ex
- Once Jupyter server has started, follow the link it prints to Docker terminal

Exercises: SQL and querying JSON objects

- PostgreSQL reference manual: https://www.postgresql.org/docs/current/static/index.html
- W3Schools SQL tutorial that covers the SQL standard but not features specific to PostgreSQL: https://www.w3schools.com/SQl/default.asp
- PostgreSQL JSON extension documentation: https://www.postgresql.org/docs/current/static/functions-json.html

- Exercise notebook name is SQL-basics

4 SQLAlchemy

- Sending SQL commands to database in raw text format has several issues:
  - Adding variables to the SQL queries requires clumsy handling of character strings
  - Handling raw character strings may make the system vulnerable to SQL injection attacks
- The Python database client used in the previous exercises, SQLAlchemy, is capable of building the SQL commands from parameters
- SQLAlchemy is not constrained to communicating with PostgreSQL and also works with several other relational databases.

Exercises: Building database queries with SQLAlchemy

- SQLAlchemy home page: HTTP://www.sqlalchemy.org
- SQLAlchemy reference manual: http://docs.sqlalchemy.org/en/rel_1_1/
- SQLAlchemy tutorials: http://www.sqlalchemy.org/library.html#tutorials
- SQLAlchemy PostgreSQL reference manual including JSON operators
  http://docs.sqlalchemy.org/en/latest/dialects/postgresql.html

- Exercise notebook name is SQLAlchemy