Data management lecture 1

What's a database?

- Storing large amounts of data in a structured way
- allows for dynamic queries for data

Databases used to be used as

- Corporate data
  - payrolls, inventory, sales, customers, accounting, documents, etc...
- Banking system
- stock exchanges
- airline systems
- etc

Database used for today

- Mobile applications
- AI
- web backends
  - web search (google, bing, etc.)
  - social networks (Facebook, Instagram, etc.)
  - blogs, forums, etc.

Why use a database?

- easy to use
- flexible search
- efficient
- multi-user access
- scalability
- secturity and consistency

Definitions

- Database Management System (DBMS) – A system that enables users to create and maintain a database.
- Database – A collection of related data. A collection of random data is not a database. Often the word database also, incorrectly, refers to a DBMS.
- Data – Known facts that can be recorded that has implicit meaning.
- Schema – A definition of table structures and their relationships.
- SQL – Structured Query Language –A language to extract data from a database.
- Table – A collection of rows and columns that contains Cells.
- Row, Tuple, or Record – A collection of cells that together form a set of data that has a concrete Relation to each other. Shown as the horizontal line to the left.
- Column, Attribute, or Field – A collection of Cells that that are all the same type. They are a part of multiple Rows
- Cell – A specific Rows Column. Ergo the intersection.
- Relation – The relationship between data in a Row.
- Value –The information saved in the specific Cell.
- View / Result Set – The table that is created in memory and sent to the client as a result of a query.

NoSQL Databases

- NoSQL means Not only SQL
- NoSQL covers multiple types of databases
  - mongoDB, Cassandra, Redis, etc

What is a relational database?

- Has a Schema that defines the Tables
- Uses multiple tables to store data
- Uses the notion of Primary Keys and Foreign keys to relate table content to each other.
- Uses SQL which makes CRUD (Create, Read, Update, Delete) operations on Schemas, Tables, and Rows.

What is SQL?

- Acronym for: Structured Query Language.
  - Pronounced either Sequel, or SQL.
- Allows for retrieval of data from a database.
- A query results in a result set, which is structured as a table with rows and columns.

Creating a Table

```
CREATE TABLE IF NOT EXISTS account(user_id serial PRIMARY KEY,
                                   username VARCHAR(50) NOT NULL,
                                   password VARCHAR(50) NOT NULL,
                                   email VARCHAR(255) NOT NULL,
                                   created_on TIMESTAMP DEFAULT NOW(),
                                   last_login TIMESTAMP DEFAULT NOW()
)
```

Inserts (CRUD)

```
INSERT INTO accounts(username,password,email) VALUES ('John', 'myPassword', 'john@acme.com');

INSERT INTO accounts(username,password,email) VALUES ('Anne', 'myPassw0rd', 'anne@acme.com');
```

Selects (CRUD)

```
SELECT * FROM accounts
```

Selecting specific data

```
SELECT * FROM accounts WHERE email = "john@acme.com";
```

Updates (CRUD)

```
UPDATE account SET password = 'newPassword' WHERE username = 'john';
```

Delete (CRUD)

```
DELETE FROM accounts WHERE email = 'john@acme.com'
```

NOTE DON'T DO THIS:

```
DELETE FROM accounts;
```

Constraints

- PRIMARY KEY – The unique identifier for the current row, which is always NOT NULL
- FOREIGN KEY – A key that refers to a primary key in a different table, signifying their relationship to each other
- UNIQUE – Two cells in this Column cannot be the same
- NOT NULL – This attribute HAS to be specified it **CANNOT** be null or undefined

PostgreSQL Data Types 1/3

- Null – Value Missing
- Boolean – 1 bit
  - Converts Boolean values e.g., 1, yes, y, t, true are converted to true, and 0, no, n false, f are converted to false.
- Character types
  - CHAR(n) –Fixed length text. Unused space is padded with space characters.
  - VARCHAR(n) – Variable length string, as in “store up to”.
  - TEXT – Large size text lengths such as book texts.

PostgreSQL Data Types 2/3

- Numeric types
  - SMALLINT $–2$ byte, ranges from -32.768 to 32.767 ($2 byte = 2 \cdot 8 bit = 256 \cdot 256 = 65.536$)
  - INT –4 byte integer, rage from $-2,147,483,648$ to $2,147,483,647$
  - SERIAL –Same as INT, but used for auto incrementing
- Floating-point
  - FLOAT(n) –floating point number with at the precision of n, and a maximum of 8 bytes. (n as in numbers after the “,”)
  - REAL –A floating point number. 0.001, 0.00000001, etc.
- Temporal types
  - DATE – dates only

- TIME – time of day values
- TIMESTAMP – both date and time values
- INTERVAL – periods of time

PostgreSQL Data Types 3/3

- UUID for storing Universally Unique Identifiers
- Array for storing array strings, numbers, etc.
- JSON stores JSON data
- hstore stores key-value pairàSpecial types
- Special types
  - box, line, point, lseg, polygon, inet, macaddr.
- see more here

Exercise

- Create Database
- Create Table
- Insert two unique Rows
- Query for all rows
- Query for the data in one of the Rows
- Delete one of the Rows
- Verify deletion by querying for all Rows again

Creating database

```
CREATE DATABASE dm01;
```

Creating table

```
CREATE TABLE IF NOT EXISTS accounts (ID SERIAL PRIMARY KEY,
                                     USERNAME VARCHAR(50) NOT NULL,
                                     PASSWORD VARCHAR(50) NOT NULL,
                                     EMAIL VARCHAR(255) NOT NULL,
                                     CREATED_ON TIMESTAMP DEFAULT NOW()
)
```

Insert two unique Rows

```
INSERT INTO accounts (USERNAME, PASSWORD, EMAIL) VALUES ('Tobias', 'Hyg57aff', 'tobias@email.com')

INSERT INTO accounts (USERNAME, PASSWORD, EMAIL) VALUES ('Jonas', '1234', 'jonas@email.com')
```

Query for all rows

```
SELECT * FROM accounts;
```

Query for the data in one of the Rows

```
SELECT * FROM accounts WHERE USERNAME='Tobias';
```

Delete one of the Rows

```
DELETE FROM accounts WHERE USERNAME='Tobias';
```

Verify deletion by querying for all Rows again

```
SELECT * FROM accounts;
```