DB2 Assignments 1

PostgreSQL Installation

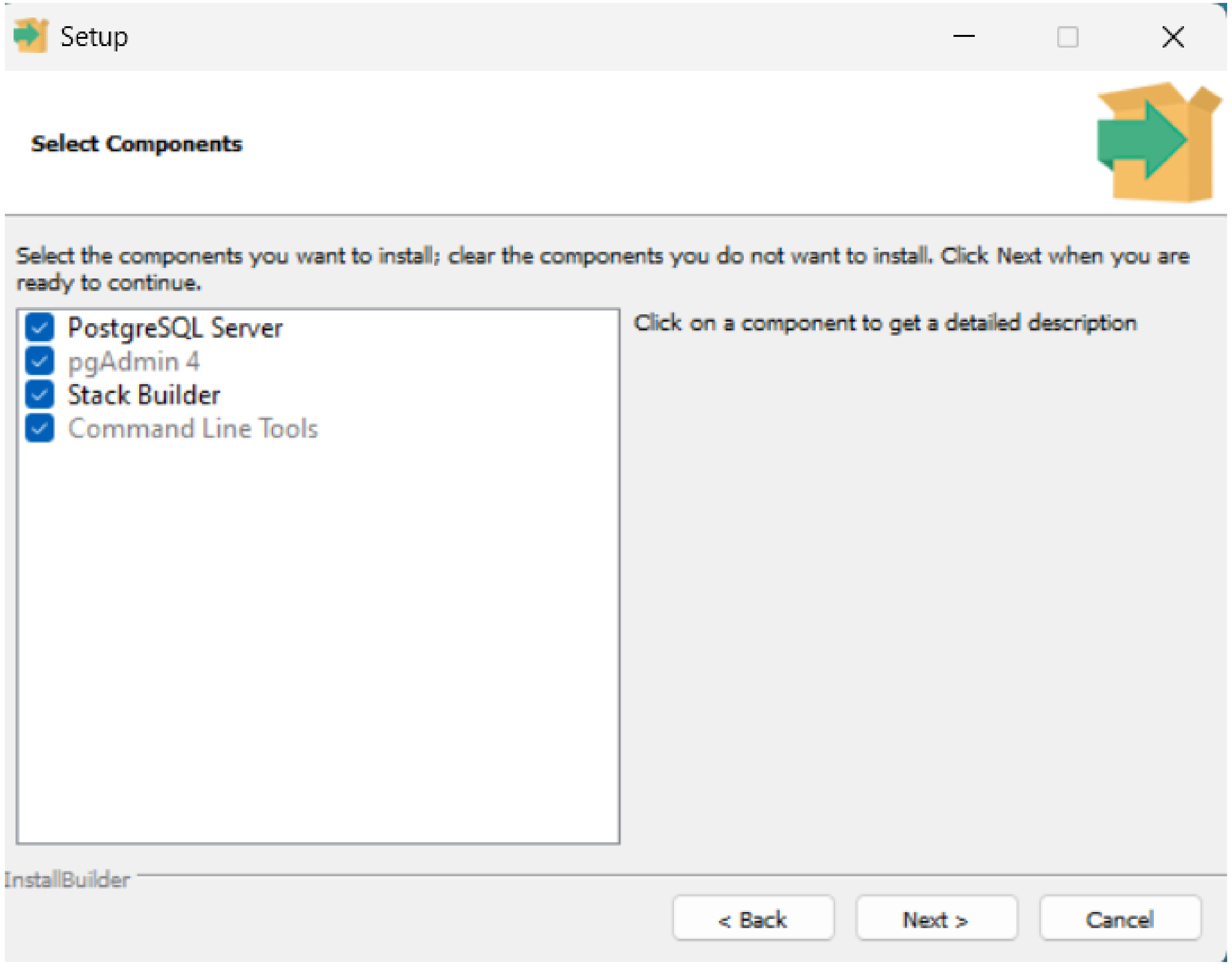
1. Download PostgresQL 16 or 17 from the PostgreSQL website https://www.postgresql.org/

2. Install together with client (pgAdmin4) and command line tool (PSQL)

3. Select storage directory, superuser password and port.

4. **Select "C" as Locale**
   If you leave it on default it takes the underlying OS locale which later may be a problem.

5. Windows: Check under services that PostgreSQL server is running.



Select Components

Select the components you want to install; clear the components you do not want to install. Click Next when you are ready to continue.

☑ PostgreSQL Server
☑ pgAdmin 4
☑ Stack Builder
☑ Command Line Tools

Click on a component to get a detailed description

PostgreSQL Installation

1. Start client pgAdmin4 and connect to your database server.

2. Open Query Tool and check your Postgres version:
   ```
   select version();
   ```

3. Select File → Preferences
   1. Nodes – set your node visibility preferences
   2. Miscelleanous – user language - English

4. Set path to PostgreSQL bin directory for command line tool PSQL, start PSQL
   ```
   psql -h localhost -p 5432 -U postgres -d <database name>
   ```

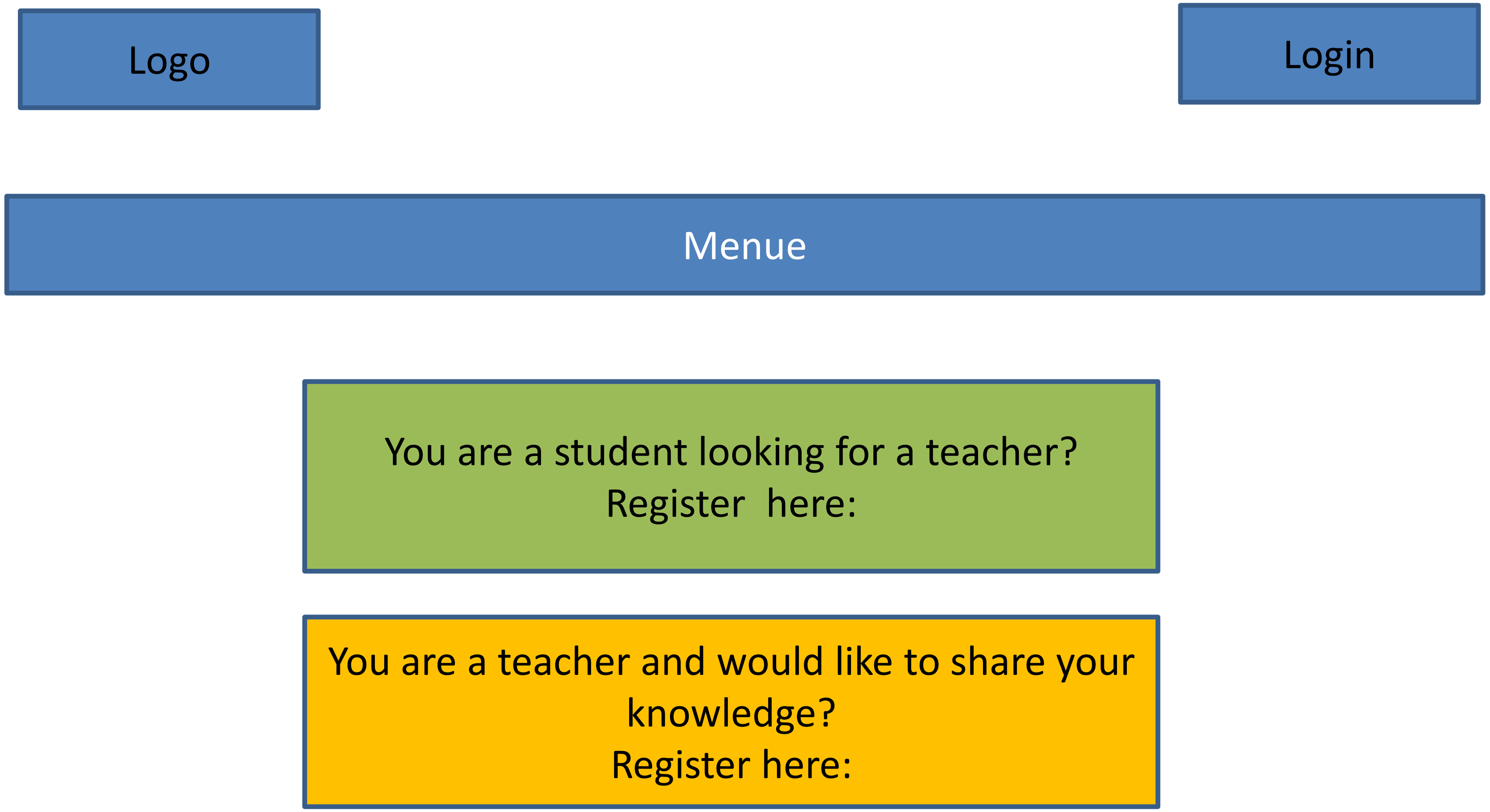
Relational Database Course Example

Business Idea: Private Lessons Agency

- Teachers and students can register on the web application. They can access their profile, read it and update it.
- The agency specifies the subjects for which it offers teachers and lessons, e.g. Math, Computer Science, Chemistry, Physics, …
- Teachers can register to offer lessons in multiple of these subjects. They can present themselves and their experience.
- Students can search for suitable teachers and buy lessons. A counter stores how many lessons a student has bought. Students can only buy lessons in increments of 5.
- Teachers also carry a counter with them. It stores how many lessons they taught and will get paid for.
- The agency, students or teachers can schedule a lesson. When a lesson is scheduled, the student's counter is decremented and the teacher's counter is incremented.
- The price for a lesson depends on the age of the student. There are 4 priceGroups: child, teen1, teen2 and adult.

- Note: **We will not be programming** this web application. We will just design, implement and use the underlying database for comparison with other databases. The database design will **NOT** represent the complete business case as we focus on areas where we can get new insights. So, please, only model and implement the given requirements.

Possible Minimal Frontend



Logo

Login

Menue

You are a student looking for a teacher?
Register here:

You are a teacher and would like to share your knowledge?
Register here:

Lesson Database

Create the **ER model** for our course database, using Chen notation.

- You will need the following entity tables: subject, teacher, student, priceType, lessonPurchase
- What relationships (relationship tables) do you need?
- Model relationships and cardinalities as demanded by the requirement
- Model the attributes as needed

Relational Reference Course Example

Which of the integrity requirements can be directly designed in the ER model and directly implmenented in the database scheme? Which of the integrity requirements need additional methods to be implemented? How would you implement these?

Referential Integrity Requirements:

1. Only registered teachers and students can schedule a lesson.
2. Teachers can register for several subjects but only for subjects that the agency offers. They can only teach the subjects that they registered for.
3. Only registered students can buy lessons. Students need to buy lessons in advance in increments of 5. A counter per student contains the number of prepaid lessons.
4. The price for a lesson depends on the student age, with 4 priceTypes: CHILD (age <10), TEEN1 (age <15), TEEN2 (age < 20) and ADULT (age 20+). This means that a lesson price for an individual student changes with his / her age.
5. Students can only schedule / take lessons that they already paid for.
6. With each lesson, the student counter is decremented by one. Teachers also have a counter. The teacher counter is incremented by one with each lesson. The teacher counter contains the teacher's current payment claim.
7. Lessons can only be scheduled in such a way that at a certain timeslot
   - a specific teacher can only teach one lesson with one student
   - a specific student can only take one lesson with one teacher
   - the teacher can only teach a lesson in a subject he registered for.

Implementation of Semantic Requirements

How can you implement the requirement that

Teachers can only teach subjects that they registered for. They can only register in subjects that the agency offers.

Implementation of Semantic Requirements

How can you implement the requirement that

Only registered students can buy lessons and they can only buy lessons in increments of 5?

Implementation of Semantic Requirements

How can you implement the requirement that

the price of a lesson depends on the age of the student?

Note that PostgresQL does NOT have generated virtual columns.

Implementation of Semantic Requirements

How can you implement the requirement that

at a certain timeslot one teacher can only teach one student in a subject that he is registered for?

PostgreSQL

Create database lesson in Pistgres. On definition tab choose:

1. Encoding: UTF-8
2. Template0
3. Strategy: WAL log
4. Locale provider: icu
5. Icu_locale: 'en-US'
6. Connection Number: -1

Relational Databases

Implement the database lesson in Postgres – as soon as in lab you agreed on an ER model that fulfills the requirements.

Relational Databases

Discuss important concepts of relational databases.

Some Information on PostgreSQL

PostgreSQL create statement

```
CREATE TABLE IF NOT EXISTS public.subject
(
    subjectcode character(2) COLLATE pg_catalog."default" NOT NULL,
    subjectname character varying(25) COLLATE pg_catalog."default" NOT NULL,
    CONSTRAINT subject_pkey PRIMARY KEY (subjectcode)
)

TABLESPACE pg_default;

ALTER TABLE IF EXISTS public.subject
    OWNER to postgres;
```

Attention: **Do NOT use capital letters** for your column names or any other identifiers because PostgreSQL expects them to be in double quotes ("). **Do NOT use CamelNotation**. Identifiers that are not in double quotes are mapped to lower case.

Autoincrement

PostgreSQL uses the data type 'serial' for **autoincrement** columns:

```
CREATE TABLE IF NOT EXISTS public.teacher
(
    t_id serial NOT NULL DEFAULT nextval('"teacher_tID_seq"'::regclass),
    t_name character varying(30) COLLATE pg_catalog."default" NOT NULL,
    t_mail character varying(50) COLLATE pg_catalog."default" NOT NULL,
    t_postalcode integer NOT NULL,
    t_dob date NOT NULL,
    t_gender gender,
    t_education education NOT NULL,
    t_remark text COLLATE pg_catalog."default",
    t_payment integer NOT NULL DEFAULT 0,
    CONSTRAINT teacher_pkey PRIMARY KEY (t_id)
)
TABLESPACE pg_default;

ALTER TABLE IF EXISTS public.teacher
    OWNER to postgres;
```

ENUM TYPES

ENUM TYPE

PostgreSQL does not have an **enum type** "out of the box". But you can create enum types yourself:

1. Create enum type gender (for teacher table and student table)
   CREATE TYPE public.gender AS ENUM ('f', 'm', 'd');
2. Create enum type education (for teacher table) that shows which education a teacher has
   Values: "Highschool", "Bachelor", "Master"
3. Check that enum types are created and visible under type in navigation bar.

Generated Columns

PostgreSQL currently implements only **stored generated columns,** no virtual generated columns.
So, Trying to calculate age with a generated column wil result in an error:

```
ALTER TABLE student
ADD COLUMN s_age integer GENERATED ALWAYS AS
(EXTRACT(YEAR FROM age('s_dob'))) stored;
```

```
ERROR: generation expression is not immutable
```

So, you need a different solution.

PostgreSQL Clusters and Databases

- A PostgreSQL database cluster contains one or more named databases. Roles and a few other object types are shared across the entire cluster. A client connection to the server can only access data in a single database, the one specified in the connection request.
- A database contains one or more named *schemas*, which in turn contain tables.
- If a table is created without specifying a schema name, by default this table is automatically put into a schema named “public”. Every new database contains such a schema. Thus, the following are equivalent:
  - CREATE TABLE IF NOT EXISTS public.teacher
  - CREATE TABLE IF NOT EXISTS teacher