IDS (Database Systems) 2021/2022

Project documentation

Task n. 40 – Web Store
(Internetový obchod)

Berezovskaia Anastasiia (xberez04), Patrik Holáň (xholan10)

Table of Contents

Task

In en:
Project n. 40
Name: Web Store.

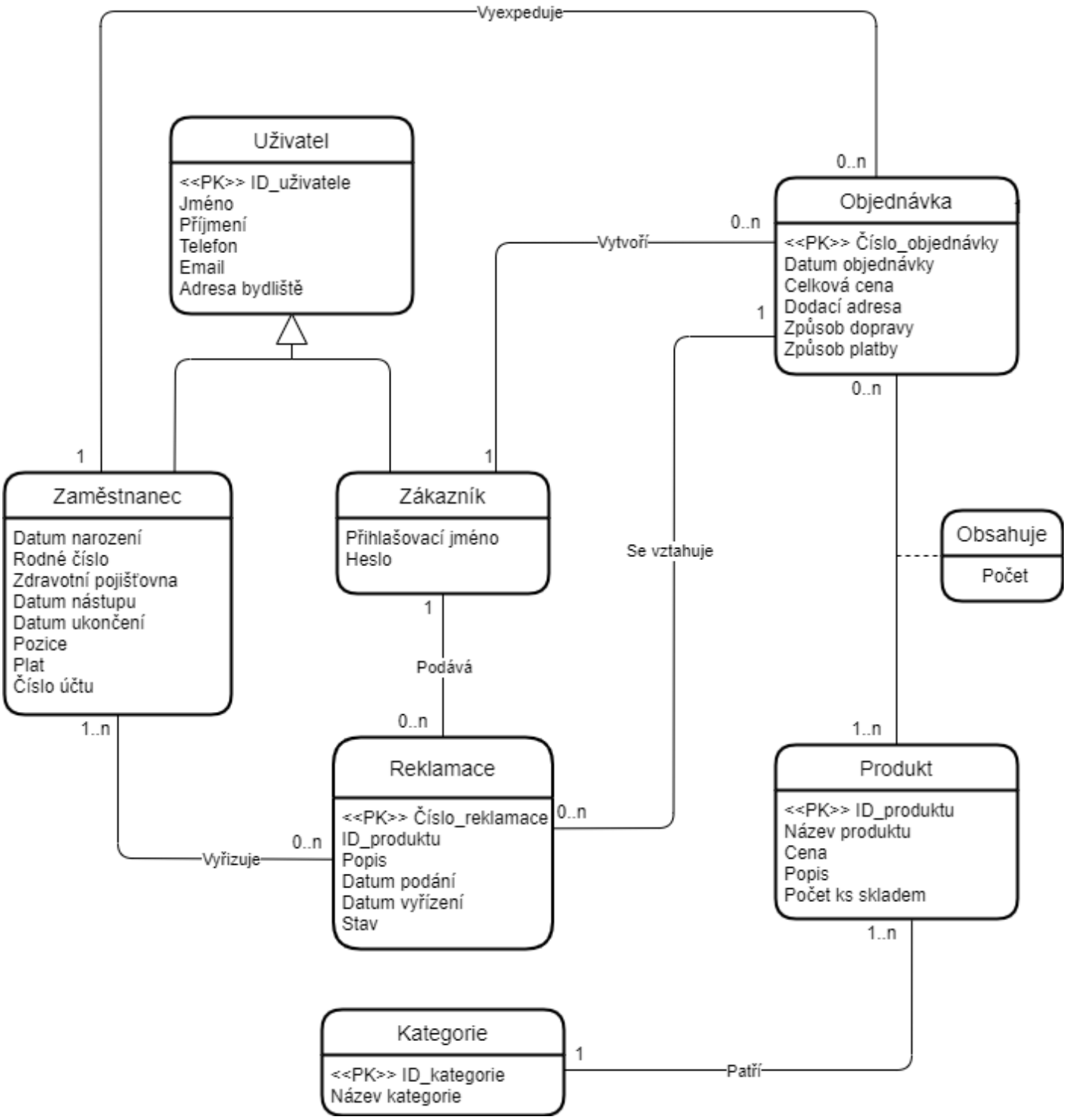
Task:
The goal is to create a simple application for an online store with a certain type of goods, such as a bookstore. Visitors to the website have the opportunity to view the entire range of stores, which is divided into categories, whether the product is in stock or not. If the visitor is interested in a certain product, he can choose it (add to shopping cart). You can order selected goods after entering the necessary data (contact, transport, ...). Only a registered user can order goods, if the user buys for the first time, he must register and obtain a login name and password. He may use this information to modify personal information. After payment by the customer for the goods, the business transaction is considered settled and the store employee ships the goods according to the order. The store management has information on total sales, popularity of goods, its capacity, orders, who handled it, etc.

In cz:
Projekt č. 40
Název projektu: Internetový obchod.

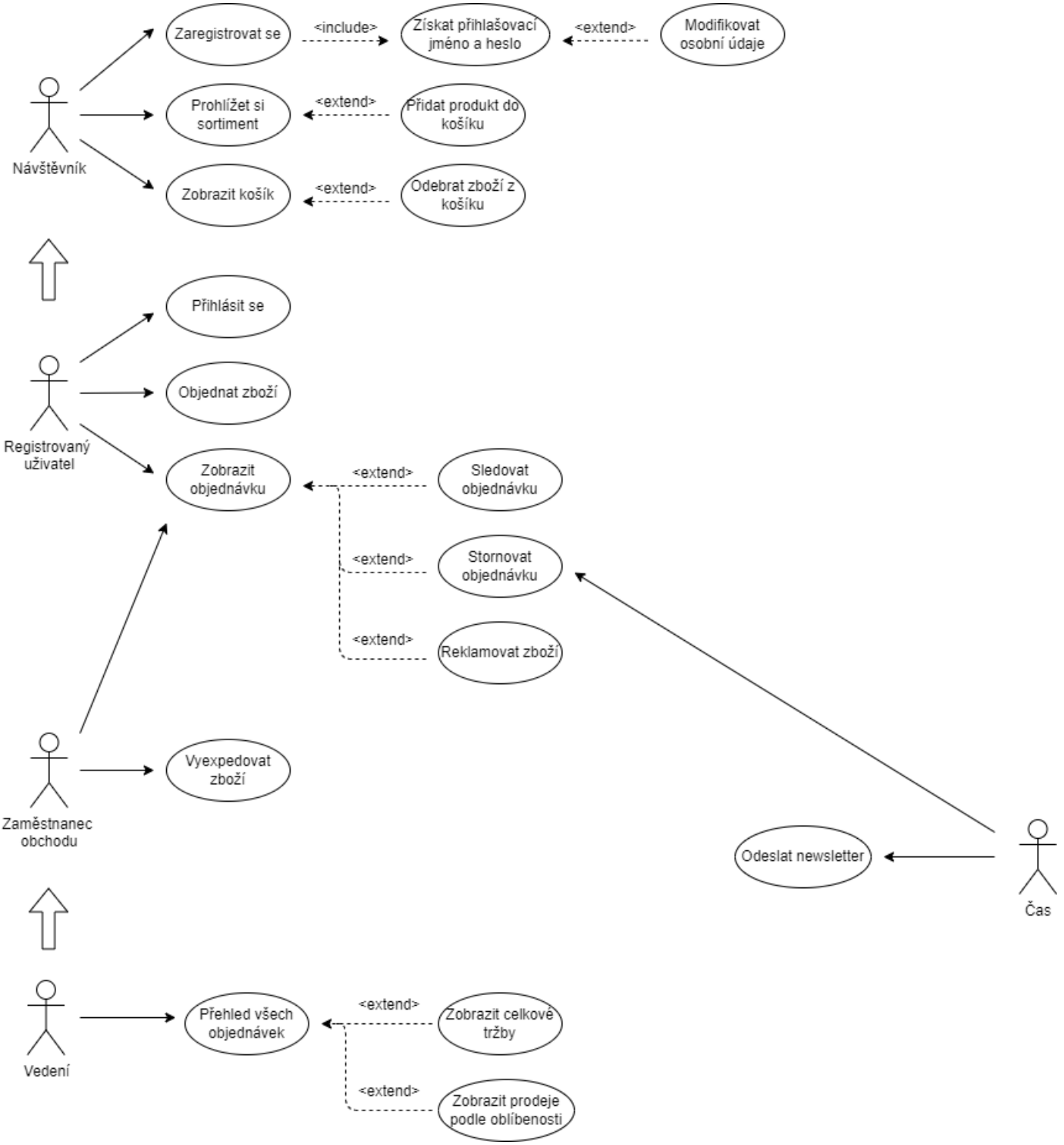
Zadání:
Cílem je vytvoření jednoduché aplikace pro internetový obchod s určitým druhem zboží např. knihkupectví. Návštěvníci WWW stránek mají možnost prohlížet si veškerý sortiment obchosu, který je členěn do kategorií, at’ už daný produkt je skladem či nikoliv. Poku má návštěvník zájem o určitý produkt, může si jej vybrat (vložení do nákupního košíku). Vybrané zboží si může objednat po zadání potřebných údajů (kontakt, doprava, ...). Zboží si může objednat pouze registrovaný uživatel, pokud uživatel nakupuje poprvé, musí se zaregistrovat a získá přihlašovácí jméno a heslo. Tyto údaje může použit k modifikaci osobních informací. Po zaplacení zákazníkem za zboží je považována obchodní transakce za vyřízenou a zaměstnanec obchodu vyexpeduje zboží podle objednávky. Vedení obchodu má informace o celkových tržbách, oblíbenosti zboží, jeho kapacitě, o objednávkách, kdo ji vyřizoval atd.

Database schema

(*Entity Relationship Diagram, Use-Case Diagram*)



ER diagram.



Use Case diagram.

Generalization/specialization

We created the generalization/specialization relationship by two entities called *employee* and *customer*, which are the specialization of the entity *username*. Each specialization is referenced to the *usernameID* attribute, which is the primary key of the *username* table.

Basic objects of database scheme

In our solution we created tables(objects):

username
customer
employee
productOrder
complaint
product
category
contains

ID format demonstration:
Login format (*username*): *+420425311677*
National ID number (*employee*): *8501254420 or 850125/4420*

Procedures

price_changes() and *percentage_field()*

The first one prints out information about price changes in the table ‘product’ before the insertion. (*Old price*: , *New price*: , *Price difference*: , *%discount* or *Price increased by*).

The second one counts percentage of fired employees (‘*employee’*). It uses *CURSOR* for the loop where variables *totalNumberOfEmployees* and *totalNumberOfFired* are incremented and as loop finishes they are used to calculate percentage of fired employees. In case there is a problem (total number of employees is 0 etc.), procedure raises errors.

Triggers

First trigger controls login format (*customer* table).
Second writes out information about price changes after *INSERT/UPDATE*.

EXPLAIN PLAN

Displays execution plan. According to the task, we implemented *EXPLAIN PLAN* joining two tables (*complaint, product*) using *COUNT()* function and *GROUP BY* query. *EXPLAIN PLAN* lists names of products that were complained and number of how many times products were complained. After *INDEX* creation searching has to quicken.

```
-----------------------------------------------------------------------------------------
| Id  | Operation                     | Name        | Rows  | Bytes | Cost (%CPU)| Time     |
-----------------------------------------------------------------------------------------
|   0 | SELECT STATEMENT              |             |     4 |   620 |     4  (25)| 00:00:01 |
|   1 |  HASH GROUP BY                |             |     4 |   620 |     4  (25)| 00:00:01 |
|   2 |   NESTED LOOPS                |             |     4 |   620 |     3   (0)| 00:00:01 |
|   3 |    NESTED LOOPS               |             |     4 |   620 |     3   (0)| 00:00:01 |
|   4 |     TABLE ACCESS FULL         | COMPLAINT   |     4 |    52 |     3   (0)| 00:00:01 |
|*  5 |     INDEX UNIQUE SCAN         | PK_PRODUCT  |     1 |       |     0   (0)| 00:00:01 |
```

After repeating the execution we got changed output:

```
-----------------------------------------------------------------------------------------
| Id  | Operation                     | Name            | Rows  | Bytes | Cost (%CPU)| Time     |
-----------------------------------------------------------------------------------------
|   0 | SELECT STATEMENT              |                 |     4 |   672 |     1   (0)| 00:00:01 |
|   1 |  NESTED LOOPS                 |                 |     4 |   672 |     1   (0)| 00:00:01 |
|   2 |   NESTED LOOPS                |                 |     4 |   672 |     1   (0)| 00:00:01 |
|   3 |    VIEW                       | VW_GBF_7        |     4 |   104 |     1   (0)| 00:00:01 |
|   4 |     HASH GROUP BY             |                 |     4 |    52 |     1   (0)| 00:00:01 |
|   5 |      INDEX FULL SCAN          | COMPLAINTNUMBER |     4 |    52 |     1   (0)| 00:00:01 |
```

MATERIALIZED VIEW

VIEW obtains list of people living in Brno.

Privilege

GRANT ALL query was used to assign rights to the second team member which allows using queries *SELECT, INSERT* etc.

GRANT EXECUTE ON grants the *EXECUTE* privilege on avgAgeEmployees procedure to the second member.