E-shop
KIV/DB1 – Semestrální práce

student: *Leonid Malakhov*
osobní číslo: *A22B0387P*
email: malakhov@students.zcu.cz
datum: 18.12.2023

1. Popis úkolu

V rámci samostatné semestrální práce vytvoří student jednoduchou databázovou
aplikaci, jejíž téma si sám zvolí. Rozsah úlohy je požadován minimálně pět
tabulek ve schématu relační databáze, přičemž tabulka typu ”číselník”není do
tohoto počtu započítávána. K vlastní úloze je vypracován referát ve struktuře
odpovídající konceptuálnímu modelování. V podstatě se skládá z:

- Popisu úlohy, ve kterém je mimo jiné uvedeno, komu (jaké roli) je úloha určena.
- Datového modelu, obvykle převzatého z použitého modelovacího nástroje a z popisu položek, u kterých nelze z jejich názvu odvodit, jaký mají význam a jakých mohou nabývat hodnot.
- Realizovaných (alespoň dvou) databázových pohledů, u kterých bude uveden jejich popis i odpovídající kód v SQL a také příslušné odpovědi nad uloženými testovacími daty.
- Řešenými scénáři, kterými lze prověřit správnost navrženého datového modelu a databázových pohledů.
- Závěru, kde je práce zhodnocena, zejména je kladně hodnocena úvaha o tom, v čem je úloha zjednodušena a proč by v předložené podobě nemohla ve skutečnosti sloužit.

2. Tabulky

Databáze “E-shop” se skládá z následujících tabulek

a. Category

Obsahuje kategorie zboží, včetně jejich názvu a popisu

b. Item

Obsahuje informace o zboží, včetně ID zboží, nazvu, popisu, ceny, dostupnosti, a ID kategorie

c. Items_List

Uchovává v sobě počet jednotlivých zboží

d. Order_table

Obsahuje informace ob objednávce, její ID, datumu generaci, statusu, useru a konečně ceně

e. Review

Obsahuje komentáře kupujících k zboží, včetně ratingu a textu

f. User

Obsahuje informace o uživatelech, jejich jméno, e-mail, adresu, datum registrace a heslo

3. SQL Dotazy

a. Dotaz 1: Seznam zboží z kategorie

Tento dotaz vrací zboží z kategorie “Electronics”. Připojuje tabulky “item” a “category”, aby našel odpovídající zboží

```
CREATE VIEW ElectronicsView AS
SELECT
    item.*,
    category.name AS category_name
FROM
    item
JOIN
    category ON item.category_category_id =
category.category_id
WHERE
    category.name = 'Electronics';
```

item_id	name	description	price	available	category_category_id
1	Smartphone	Latest model with advanced features	800	Y	1
2	Laptop	Powerful laptop for professional use	1200	Y	1

b. Dotaz 2: Počet zboží každé kategorie

Tento dotaz vrací počet zboží z každé kategorie.

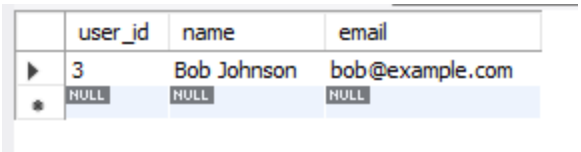
```
CREATE VIEW CategoryItemsCount AS
SELECT
    cat.name AS Name,
    COUNT(item.item_id) AS "Items count"
FROM
    category cat
JOIN
    item ON item.category_category_id = cat.category_id
GROUP BY
    cat.name;
```

	Name	Items count
▶	Books	1
	Clothing	2
	Electronics	2

c. Dotaz 3: Seznam uživatelů bez objednávek

Tento dotaz vrací seznam uživatelů, kteří ještě nemají objednávku.

```
CREATE VIEW UsersWithoutOrders AS
SELECT
    user_id,
    name,
    email
FROM
    user
WHERE
    user_id NOT IN (SELECT DISTINCT user_user_id FROM
order_table);
```



user_id	name	email
3	Bob Johnson	bob@example.com
NULL	NULL	NULL

4. Scenare

a. Scenar 1: Přidání testovací objednávky uživatelem

Tento scénář ukazuje, jak lze v databází přidat testovacího uživatele a následně vygenerovat objednávku z obsahem ze zboží.

```
START TRANSACTION;
SAVEPOINT SAVE;

SET @user_id = 666;

INSERT INTO user (user_id, name, email, address,
registration_date, password) VALUES
(@user_id, 'Den Denov', 'den@example.com', '66 Second St',
"2023-12-18", 123456);

SET @order_id = 666;

INSERT INTO order_table (order_id, date_creation, status,
user_user_id, total_cost) VALUES
(@order_id, "2023-12-19", 'Pending', @user_id, NULL);

INSERT INTO items_list (items_list_id, order_order_id,
item_item_id, amount) VALUES
(111, @order_id, 1, 2),
(222, @order_id, 4, 1);



SET @total_price = (
    SELECT SUM(item.price * items_list.amount)
    FROM item
    JOIN items_list ON items_list.item_item_id = item.item_id
```

```
    WHERE items_list.order_order_id = @order_id
);

UPDATE order_table
SET total_cost = @total_price
WHERE order_table.order_id = @order_id;

SELECT *
FROM order_table ord
WHERE ord.order_id = @order_id;

ROLLBACK TO SAVE;
```

order_id	date_creation	status	user_user_id	total_cost
666	2023-12-19	Pending	666	1650
NULL	NULL	NULL	NULL	NULL

order_table 4

Output

Action Output

#	Time	Action
1	19:53:02	SELECT * FROM item JOIN category cat ON item.category_category_id = cat.category_id WHERE cat.name = 'Electronics' LIMIT 0, 1000
2	19:57:50	SELECT cat.name AS 'Name', COUNT(item.item_id) AS 'Items count' FROM category cat JOIN item ON item.category_category_id = cat.category_id GROUP BY cat.name LIMIT 0, 1000
3	20:00:08	SELECT user_id, name, email FROM user WHERE user_id NOT IN (SELECT DISTINCT user_user_id FROM order_table) LIMIT 0, 1000
4	20:04:38	START TRANSACTION
5	20:04:38	SAVEPOINT SAVE
6	20:04:38	SET @user_id = 666
7	20:04:38	INSERT INTO user (user_id, name, email, address, registration_date, password) VALUES (@user_id, 'Den Denov', 'den@example.com', '66 Second St', ''2023-12-18'', 123456)
8	20:04:38	SET @order_id = 666
9	20:04:38	INSERT INTO order_table (order_id, date_creation, status, user_user_id, total_cost) VALUES (@order_id, ''2023-12-19'', 'Pending', @user_id, NULL)
10	20:04:38	INSERT INTO items_list (items_list_id, order_order_id, item_item_id, amount) VALUES (111, @order_id, 1, 2), (222, @order_id, 4, 1)
11	20:04:38	SET @total_price = (SELECT SUM(item.price * items_list.amount) FROM item JOIN items_list ON items_list.item_item_id = item.item_id WHERE items_list.order_order_id = @order_id)
12	20:04:38	UPDATE order_table SET total_cost = @total_price WHERE order_table.order_id = @order_id
13	20:04:38	SELECT * FROM order_table ord WHERE ord.order_id = @order_id LIMIT 0, 1000
14	20:04:38	ROLLBACK TO SAVE

b. Scenar 2: Aktualizace ceny zboží a sledování změn

Tento scénář ukazuje na průběh aktualizace ceny zboží.

```
START TRANSACTION;
SAVEPOINT SAVE;

SELECT item_id, name, description, price, available FROM item
ORDER BY price DESC;

UPDATE item SET price = price - 250 WHERE item_id = 2;
```

```
UPDATE item SET price = price + 200 WHERE item_id = 1;

SELECT item_id, name, description, price, available FROM item
ORDER BY price DESC;

ROLLBACK TO SAVE;
```

item_id	name	description	price	available
2	Laptop	Powerful laptop for professional use	1200	Y
1	Smartphone	Latest model with advanced features	800	Y
4	Jeans	Classic denim jeans	50	Y
3	T-shirt	Cotton T-shirt for casual wear	20	Y
5	Book	Bestseller novel	15	Y

item_id	name	description	price	available
1	Smartphone	Latest model with advanced features	1000	Y
2	Laptop	Powerful laptop for professional use	950	Y
4	Jeans	Classic denim jeans	50	Y
3	T-shirt	Cotton T-shirt for casual wear	20	Y
5	Book	Bestseller novel	15	Y

5. Zaver

Během tohoto semestru jsem osobně provedl rozsáhlou analýzu, návrh a implementaci databáze pro internetový obchod. V průběhu

vytváření datového modelu jsem se sám zaměřil na správnou definici vztahů mezi tabulkami, což umožnilo efektivní ukládání a manipulaci s daty. Dále jsem vytvořil skripty pro vytvoření databáze, naplnil ji testovacími daty a vytvořil pohledy, které poskytují užitečné informace o objednávkách a uživatelských preferencích.

V průběhu implementace jsem čelil různým výzvám, zejména při správě cizích klíčů a optimalizaci dotazů pro získání přesných výsledků. Přesto jsem tyto výzvy překonal a dosáhl jsem efektivního a spolehlivého řešení pro správu dat internetového obchodu.

Tato práce mi poskytla hlubší porozumění konceptů relačních databází a praktických aspektů návrhu a implementace databázových systémů. Věřím, že vytvořená databáze bude solidním základem pro budoucí rozšíření a optimalizaci, aby mohla efektivně podporovat rostoucí potřeby internetového obchodu a zlepšovat uživatelský zážitek.