# MySQL Exam Online store – electronic devices

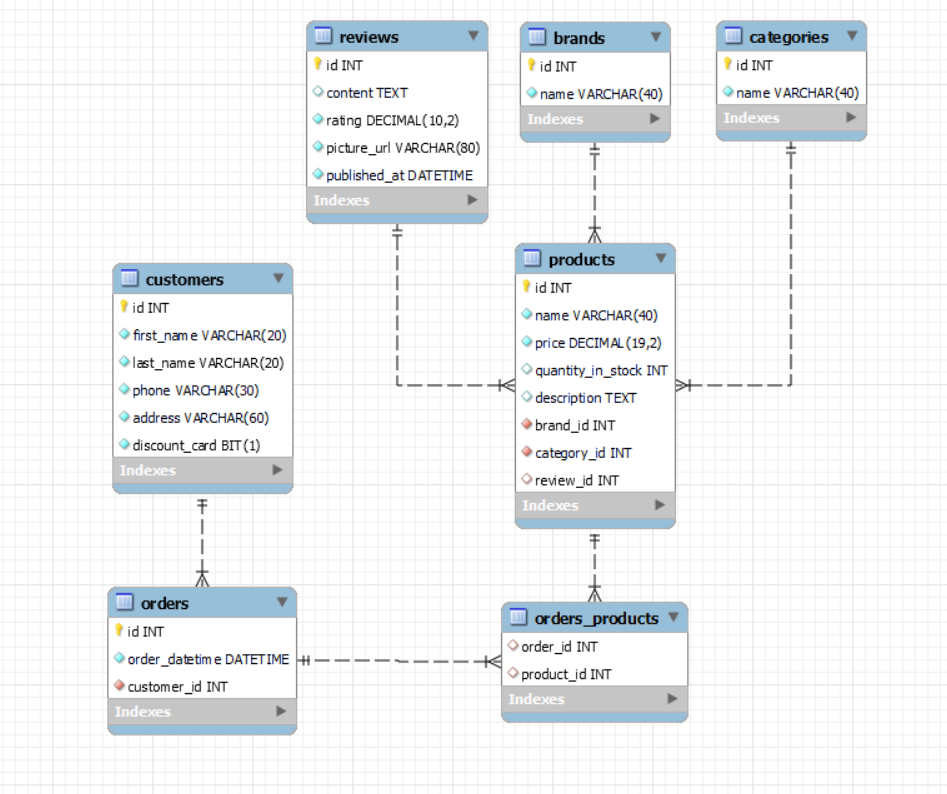
[Link for Database Basics MySQL Exam - 13 February 2022](https://judge.softuni.org/Contests/3314/Database-Basics-MySQL-Exam-13-February-2022)

A client who owns a big online store for electronic devices needs your help to manage the database of the shop.

Help him by implementing the database structure, optimize his system and make analysis for the future management strategy of the shop.

## Section 0: Database Overview

You have been given an Entity / Relationship Diagram of the Database:



The **online\_stores’s** **Database** needs to hold information about **products**, **customers**, **orders**, **categories**, **brands, reviews**.

Your task is to create a database called **online\_store**. Then you will have to create several **tables**.

* brands – contains information about the **brands**.
* categories – contains information about the **categories**.
* reviews – contains information about the **reviews**.
* products – contains information about the **products**.
  + Each product has a brand, **review** and **category**.
* customers – contains information about the **customers**.
* orders – contains information about the **orders**.
  + Each order has a customer.
* orders\_products – a **many** to **many** **mapping** table between the **orders** and the **products**.

## Section 1: Data Definition Language (DDL) – 40 pts

Make sure you implement the whole database correctly on your local machine, so that you could work with it.

The instructions you'll be given will be the minimal needed for you to implement the database.

### Table Design

You have been tasked to create the tables in the database by the following models:

**brands**

|  |  |  |
| --- | --- | --- |
| **Column Name** | **Data Type** | **Constraints** |
| id | **Integer,** from **1** to **2,147,483,647.** | **Primary Key AUTO\_INCREMENT** |
| name | A **string** containing a maximum of **40 characters**. Unicode is **NOT** needed. | **NULL** is **NOT** permitted**. UNIQUE** values. |

**categories**

|  |  |  |
| --- | --- | --- |
| **Column Name** | **Data Type** | **Constraints** |
| id | **Integer,** from **1** to **2,147,483,647.** | **Primary Key AUTO\_INCREMENT** |
| name | A **string** containing a maximum of **40 characters**. Unicode is **NOT** needed. | **NULL** is **NOT** permitted**. UNIQUE** values. |

**reviews**

|  |  |  |
| --- | --- | --- |
| **Column Name** | **Data Type** | **Constraints** |
| id | **Integer,** from **1** to **2,147,483,647.** | **Primary Key AUTO\_INCREMENT** |
| content | A very **long** string field |  |
| rating | **DECIMAL**, up to **10 digits**, **2** of which after the **decimal point**. | **NULL** is **NOT** permitted**.** |
| picture\_url | A **string** containing a maximum of **80 characters**. Unicode is **NOT** needed. | **NULL** is **NOT** permitted**.** |
| published\_at | The publishing **datetime** of the review. | **NULL** is **NOT** permitted**.** |

**products**

|  |  |  |
| --- | --- | --- |
| **Column Name** | **Data Type** | **Constraints** |
| id | **Integer,** from **1** to **2,147,483,647.** | **Primary Key AUTO\_INCREMENT** |
| name | A **string** containing a maximum of **40 characters**. Unicode is **NOT** needed. | **NULL** is **NOT** permitted**.** |
| price | **DECIMAL**, up to **19 digits**, **2** of which after the **decimal point**. | **NULL** is **NOT** permitted**.** |
| quantity\_in\_stock | **Integer,** from **1** to **2,147,483,647.** |  |
| description | A very **long** string field |  |
| brand\_id | **Integer,** from **1** to **2,147,483,647.** | Relationship with table brands.  **NULL** is **NOT** permitted**.** |
| category\_id | **Integer,** from **1** to **2,147,483,647.** | Relationship with table categories.  **NULL** is **NOT** permitted**.** |
| review\_id | **Integer,** from **1** to **2,147,483,647.** | Relationship with table reviews. |

**customers**

|  |  |  |
| --- | --- | --- |
| **Column Name** | **Data Type** | **Constraints** |
| id | **Integer,** from **1** to **2,147,483,647.** | **Primary Key AUTO\_INCREMENT** |
| first\_name | A **string** containing a maximum of **20 characters**. Unicode is **NOT** needed. | **NULL** is **NOT** permitted**.** |
| last\_name | A **string** containing a maximum of **20 characters**. Unicode is **NOT** needed. | **NULL** is **NOT** permitted**.** |
| phone | A **string** containing a maximum of **30 characters**. Unicode is **NOT** needed | **NULL** is **NOT** permitted**. UNIQUE** values. |
| address | A **string** containing a maximum of **60 characters**. Unicode is **NOT** needed. | **NULL** is **NOT** permitted**.** |
| discount\_card | Can be true or false | **NULL** is **NOT** permitted**.**  **Default** is **FALSE** |

**orders**

|  |  |  |
| --- | --- | --- |
| **Column Name** | **Data Type** | **Constraints** |
| id | **Integer,** from **1** to **2,147,483,647.** | **Primary Key AUTO\_INCREMENT** |
| order\_datetime | The publishing **datetime** of the order. | **NULL** is **NOT** permitted**.** |
| customer\_id | **Integer,** from **1** to **2,147,483,647.** | Relationship with table customers.  **NULL** is **NOT** permitted**.** |

**orders\_products**

|  |  |  |
| --- | --- | --- |
| **Column Name** | **Data Type** | **Constraints** |
| order\_id | **Integer**, from **1** to **2,147,483,647.** | Relationship with table orders. |
| product\_id | **Integer**, from **1** to **2,147,483,647.** | Relationship with table products. |

Submit your solutions in Judge on the first task. Submit **all** SQL table creation statements.

You will also be given a data.sql file. It will contain a **dataset** with random data which you will need to **store** in your **local database**. This data will be given to you so you will not have to think of data and lose essential time in the process. The data is in the form of **INSERT** statement queries.

## Section 2: Data Manipulation Language (DML) – 30 pts

Here we need to do several manipulations in the database, like changing data, adding data etc.

### Insert

You will have to **insert** records of data into the **reviews** table, based on the **products** table.

For **products** with **id** equal or greater than **5**, **insert data** in the **reviews** table with the **following values**:

• **content** – set it to the first **15** characters from the **description** of the **product**.

• **picture\_url** – set it to the **product's** name but **reversed**.

• **published\_at** – set it to **10-10-2010**.

• **rating** – set it to the **price** of the **product** divided by **8**.

### Update

**Reduce** all **products** **quantity** by 5 for **products** with **quantity** equal to or greater than 60 and less than 70 (**inclusive**).

### Delete

**Delete** all **customers**, who didn't order anything.

## Section 3: Querying – 50 pts

And now we need to do some data extraction. **Note** that the **example results** from **this section** use a **fresh database**. It is **highly recommended** that you **clear** the **database** that has been **manipulated** by the **previous problems** from the **DML** **section** and **insert again** the **dataset** you’ve been given, to ensure **maximum consistency** with the **examples** given in this section.

### Categories

Extract from the **online\_store** system database, info about the name of **categories**.

**Order** the results by **category\_name** in **descending** order;

#### Required Columns

* id (categories)
* name

#### Example

|  |  |
| --- | --- |
| **id** | **name** |
| 5 | Small domestic appliances |
| ... | ... |

### Quantity

Write a query that returns: **product\_id, brand\_id**, **name** and **quantity** from **table** products. **Filter** products which **price** is higher than 1000 and their **quantity** is lower than 30.

**Order** the results **ascending** by **quantity\_in\_stock**, then by **id**.

#### Required Columns

* id (product)
* brand\_id
* name (product)
* quantity\_in\_stock

#### Example

|  |  |  |  |
| --- | --- | --- | --- |
| **id** | **brand\_id** | **name** | **quantity\_in\_stock** |
| 19 | 6 | GeneralElectric superCool | 4 |
| 36 | 3 | Overhold | 13 |
| 2 | 11 | DSLR camera NIKON D5600 | 21 |
| 27 | 3 | Eagle Vision GS340 | 24 |

### Review

Write a query that returns: **id, content**, **rating,** **picture\_url** and **published\_at** for all reviews which **content** starts with **‘My’** and the characters of the content are more than 61 symbols.

**Order** by rating in **descending** order.

#### Required Columns

* id (reviews)
* content
* rating
* picture\_url
* published\_at

#### Example

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **id** | **content** | **rating** | **picture\_url** | **published\_at** |
| 13 | My kids love this product! I am amazed how well it works actually. | 4.80 | kjF54Foiu982 | 2020-10-12 23:12:00 |
| 12 | My experience with those type of products is really bad, but i am very impressed with this | 4.70 | ds245asdfa84e88 | 2019-12-11 13:14:00 |
| 2 | My neighbor Karly has one of these. She works as a gambler and she says it looks tall. | 4.60 | ukQQ\_TEYAfBGkw%3A1635 | 2019-10-02 14:32:00 |
| … | … | … | … | … |

### First customers

There are many customers in our shop system, but we need to find only those who are clients from the beginning of the online store creation.

Extract from the database, the **full name** of **customer**, the **address**, and the **date of order**. The **year** must be lower or equal to 2018.

**Order** the results **descending** by **full\_name.**

#### Required Columns

* full\_name (first\_name + " " + last\_name)
* address
* order\_datetime

#### Examples

|  |  |  |
| --- | --- | --- |
| **full\_name** | **address** | **order\_date** |
| Shirley Clayfield | 51 East Drive | 2018-02-04 20:34:33 |
| Selene Mateo | 9 Elka Point | 2018-04-13 07:44:15 |
| … | … | … |
| Cherilyn Bradane | 323 Steensland Parkway | 2018-12-03 05:19:24 |

### Best categories

Extract from the database, the **top 5 categories**.

**Order** the results **descending** by **items count**, then by **total\_quantity** in **ascending**.

#### Required Columns

* items\_count (number of items in the category)
* name
* total\_quantity (sum of all items in this category)

#### Example

|  |  |  |
| --- | --- | --- |
| **items\_count** | **name** | **total\_quantity** |
| 12 | Small domestic appliances | 639 |
| 10 | Phones and tablets | 775 |
| 9 | Photo and Video | 259 |
| 9 | Laptops | 437 |
| 9 | Home appliances | 542 |

## Section 4: Programmability – 30 pts

The time has come for you to prove that you can be a little more dynamic on the database. So, you will have to write several procedures.

### Extract client cards count

Create a **user defined function** with the name **udf\_customer\_products\_count(name VARCHAR(30))** that receives a **customer’s first name** and returns the total number of products he ordered;

#### Required Columns

* first\_name (customers)
* last\_name (customers)
* total\_products (udf\_customer\_products\_count)

#### Example

|  |  |  |
| --- | --- | --- |
| **Query** | | |
| SELECT c.first\_name,c.last\_name, udf\_customer\_products\_count('Shirley') as `total\_products` FROM customers c  WHERE c.first\_name = 'Shirley'; | | |
| first\_name | last\_name | total\_products |
| Shirley | Clayfield | 5 |

### Reduce price

Create a stored procedure udp\_reduce\_price which accepts the following parameters:

* category\_name (VARCHAR(50))

Extracts data about the **products** from the given **category** and reduces the **prices** by **30%** of all **products** which have **reviews** with **rating** less than **4** and are from the given **category**.

#### Result

|  |
| --- |
| **Query** |
| CALL udp\_reduce\_price ('Phones and tablets'); |
| This execution will update 6 products – Galaxy M52 5G, Galaxy Z Fold3 5G, XS 420 … |
| Result |
| Galaxy M52 5G - 789.00 -> 552.30  Galaxy Z Fold3 5G – 599.00 -> 419.30  XS 420 – 754.00 -> 527.80  … … … |