# Database Basics MS SQL Regular Exam – 07 Aug 2024

Exam problems for the ["Database Basics" course @ SoftUni](https://softuni.bg/trainings/4534/ms-sql-may-2024).  
Submit your solutions in the SoftUni Judge system at [Judge](https://judge.softuni.org/Contests/4823/MS-SQL-Retake-Exam-7-August-2024).

# Shoes Application Database

# Section 1. DDL (30 pts)

You have been given the E/R Diagram of the **ShoesApplicationDatabase**



Create a database called **ShoesApplicationDatabase**. You need to create **6 tables**:

* **Users** - Contains information about each user in the датабасе;
* **Brands** - Contains information about the brands that offer shoes;
* **Sizes** - Contains information about shoe sizes;
* **Shoes** - Contains information about the shoes available;
* **Orders** - Contains information about the orders placed by users;
* **ShoesSizes** - Мanages the **many-to-many** relationship between **shoes** and **sizes**,   
  indicating which sizes are available for specific shoes;

**NOTE: Keep in mind that Judge doesn't accept "ALTER" statements and square brackets naming (when the names are not keywords).**

**NOTE: Please keep in mind that in case you have to work with a date, you have to use the same data type, described in the models tables. If you don't use the correct type, the Judge system won't accept your submission as correct.**

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

### Users

|  |  |  |
| --- | --- | --- |
| **Column Name** | **Data Type** | **Constraints** |
| **Id** | **Integer** from **0** to **2,147,483,647** | PK, Unique table identification, Identity |
| **Username** | **String** up to **50** symbols, **Unicode** | **Unique, Null** is **not** allowed |
| **FullName** | **String** up to **100** symbols, **Unicode** | **Null** is **not** allowed |
| **PhoneNumber** | **String** up to **15** symbols, **Unicode** | **Null** is allowed |
| **Email** | **String** up to **100** symbols, **Unicode** | **Unique, Null** is **not** allowed |

### Brands

|  |  |  |
| --- | --- | --- |
| **Column Name** | **Data Type** | **Constraints** |
| **Id** | **Integer** from **0** to **2,147,483,647** | PK, Unique table identification, Identity |
| **Name** | **String** up to **50** symbols, **Unicode** | **Unique, Null** is **not** allowed |

### Sizes

|  |  |  |
| --- | --- | --- |
| **Column Name** | **Data Type** | **Constraints** |
| **Id** | **Integer** from **0** to **2,147,483,647** | PK, Unique table identification, Identity |
| **EU** | **Decimal**, up to **5 digits**, **2** of which after the **decimal point** | **Null** is **not** allowed |
| **US** | **Decimal**, up to **5 digits**, **2** of which after the **decimal point** | **Null** is **not** allowed |
| **UK** | **Decimal**, up to **5 digits**, **2** of which after the **decimal point** | **Null** is **not** allowed |
| **CM** | **Decimal**, up to **5 digits**, **2** of which after the **decimal point** | **Null** is **not** allowed |
| **IN** | **Decimal**, up to **5 digits**, **2** of which after the **decimal point** | **Null** is **not** allowed |

### ****Shoes****

|  |  |  |
| --- | --- | --- |
| **Column Name** | **Data Type** | **Constraints** |
| **Id** | **Integer** from **0** to **2,147,483,647** | PK, Unique table identification, Identity |
| **Model** | **String** up to **30** symbols, **Unicode** | **Null** is **not** allowed |
| **Price** | **Decimal**, up to **10 digits**, **2** of which after the **decimal point** | **Null** is **not** allowed |
| **BrandId** | **Integer** from **0** to **2,147,483,647** | PK, Unique table identification, Relationship with table **Brands**,  **Null** is not allowed |

### Orders

|  |  |  |
| --- | --- | --- |
| **Column Name** | **Data Type** | **Constraints** |
| **Id** | **Integer** from **0** to **2,147,483,647** | PK, Unique table identification, Identity |
| **ShoeId** | **Integer** from **0** to **2,147,483,647** | PK, Unique table identification, Relationship with table **Shoes**,  **Null** is not allowed |
| **SizeId** | **Integer** from **0** to **2,147,483,647** | PK, Unique table identification, Relationship with table **Sizes**,  **Null** is not allowed |
| **UserId** | **Integer** from **0** to **2,147,483,647** | PK, Unique table identification, Relationship with table **Users**,  **Null** is not allowed |

### ****ShoesSizes****

|  |  |  |
| --- | --- | --- |
| **Column Name** | **Data Type** | **Constraints** |
| **ShoeId** | **Integer** from **0** to **2,147,483,647** | PK, Unique table identification, Relationship with table **Shoes**, **Null** is not allowed |
| **SizeId** | **Integer** from **0** to **2,147,483,647** | PK, Unique table identification, Relationship with table **Sizes**, **Null** is not allowed |

## Database design

Submit all of your **CREATE** **statements** to Judge (only the creation of tables).

# Section 2. DML (10 pts)

**Before you start, you have to import "Dataset.sql". If you have created the structure correctly, the data should be successfully inserted.**

In this section, you have to do some data manipulations:

## Insert

Let's **insert** some sample data into the database. Write a query to add the following records to the corresponding tables. All IDs (**Primary Keys**) should be **auto-generated**.

### ****Brands****

|  |
| --- |
| **Name** |
| 'Timberland' |
| 'Birkenstock' |

### ****Shoes****

|  |  |  |
| --- | --- | --- |
| **Model** | **Price** | **BrandId** |
| 'Reaxion Pro' | 150.00 | 12 |
| 'Laurel Cort Lace-Up' | 160.00 | 12 |
| 'Perkins Row Sandal' | 170.00 | 12 |
| 'Arizona' | 80.00 | 13 |
| 'Ben Mid Dip' | 85.00 | 13 |
| 'Gizeh' | 90.00 | 13 |

### ****ShoesSizes****

|  |  |  |  |
| --- | --- | --- | --- |
| **ShoeId** | **SizeId** | **ShoeId** | **SizeId** |
| 70 | 1 | 73 | 1 |
| 70 | 2 | 73 | 3 |
| 70 | 3 | 73 | 5 |
| 71 | 2 | 74 | 2 |
| 71 | 3 | 74 | 4 |
| 71 | 4 | 74 | 6 |
| 72 | 4 | 75 | 1 |
| 72 | 5 | 75 | 2 |
| 72 | 6 | 75 | 3 |

### ****Orders****

|  |  |  |
| --- | --- | --- |
| **S**hoe**Id** | **S**ize**Id** | **U**serId |
| 70 | 2 | 15 |
| 71 | 3 | 17 |
| 72 | 6 | 18 |
| 73 | 5 | 4 |
| 74 | 4 | 7 |
| 75 | 1 | 11 |

## Update

Write an **SQL query to update the price of all shoes from the brand 'Nike' by increasing them by 15%.** Make sure to **identify the brand** correctly using its name and **apply the price adjustment** to **all corresponding records** in the **Shoes** table.

## Delete

Write an SQL query to **delete a specific shoe model** from the Shoes table because the brand has stopped selling it. Be aware that **deleting this shoe model might cause referential integrity conflicts with related tables**. Your task is to **ensure that all related records are appropriately handled** to maintain database integrity.

Let's assume we are **deleting** the shoe with **Model = 'Joyride Run Flyknit'**

# Section 3. Querying (40 pts)

**You need to start with a fresh dataset, so recreate your DB and import the sample data again ("Dataset.sql**"**).**

## Orders By Price of the Shoe

Select all **orders,** ordered by **price** of the shoe – **descending**, and then by the **shoe model**, alphabetically (**ascending**).

Required columns:

* **ShoeModel**
* **Price**

### Example

|  |  |
| --- | --- |
| **ShoeModel** | **Price** |
| 'Alphaedge 4D' | 300.00 |
| ' ZoomX Vaporfly' | 250.00 |
| 'Reebok Legacy Lifter II' | 200.00 |
| ' Joyride Run Flyknit' | 180.00 |
| 'Ultraboost 21' | 180.00 |
| 'React Infinity Run' | 160.00 |
| … | … |

## Shoes With Their Brand

Select **all shoe models** along with their **respective brands**. **Order** the results **alphabetically** by **brand name** and **then** by **shoe model**.

Required columns:

* **BrandName**
* **ShoeModel**

### Example

|  |  |
| --- | --- |
| **BrandName** | **ShoesCount** |
| Adidas | Adizero Adios |
| Adidas | Alphaedge 4D |
| Adidas | Continental 80 |
| Adidas | Gazelle |
| Adidas | NMD\_R1 |
| … | … |

## Top 10 Users By Total Money Spent

Select the **top 10 users** who have **spent the most money**. Extract **UserId**, **FullName**, and **TotalSpent**. Order the result by **TotalSpent descending**, then by **FullName, alphabetically** (ascending).

Required columns:

* **UserId**
* **FullName**
* **TotalSpent**

### Example

|  |  |  |
| --- | --- | --- |
| **UserId** | **FullName** | **TotalSpent** |
| 7 | 'Frank Jackson' | 505.00 |
| 20 | 'Sam Stewart' | 470.00 |
| 12 | 'Karen Moore' | 430.00 |
| … | … | … |

## Average Price Of Orders

Extract information about **all users** with their **usernames**, **emails**, and the **average price of their orders**. Only consider **users with more than 2 orders**. **Order** the results **by the average price** in **descending** order.

**The average price must be rounded to two decimal places and trailing zeros removed.**

Required columns:

* **Username**
* **Email**
* **AvgPrice**

### Example

|  |  |  |
| --- | --- | --- |
| **Username** | **Email** | **AvgPrice** |
| 'fjackson' | 'fjackson@example.com' | 168.33 |
| 'sstewart' | 'sstewart@example.com' | 156.67 |
| 'kmoore' | 'kmoore@example.com' | 143.33 |
| … | … | … |

## Running Shoes

You need to write a query that extracts **all Nike shoe models** with **more than five different sizes available** and **contains the word 'Run' in the model name**. The **results should include the shoe model**, **the count of different sizes** available for each model, and **the brand name**. Finally, the results should be **ordered by the model name** in **descending alphabetical order**.

Required columns:

* **Model**
* **CountOfSizes**
* **BrandName**

### Example

|  |  |  |
| --- | --- | --- |
| **Model** | **CountOfSizes** | **BrandName** |
| 'React Infinity Run' | 6 | 'Nike' |
| 'Joyride Run Flyknit' | 6 | 'Nike' |

## Phone Including Digits 345

Your task is to write a query to **extract full information about all shoes** ordered by **users whose phone numbers contain '345'** in the shoe application database. You need to select the **user's full name**, **phone number**, the **price of the order**, the **shoe id**, the **brand id**, and the **shoe size formatted in EU/US/UK**. The results should be **ordered by the shoe model name in alphabetical order**.

Required columns:

* **FullName**
* **PhoneNumber**
* **OrderPrice**
* **ShoeId**
* **BrandId**
* **ShoeSize => Format Example:** '45.00EU/10.00US/7.50UK'

### Example

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **FullName** | **PhoneNumber** | **OrderPrice** | **ShoeId** | **BrandId** | **ShoeSize** |
| 'Sam Stewart' | '555-012-3459' | 80.00 | 23 | 3 | '45.00EU/10.00US/7.50UK' |
| 'Bob White' | '555-345-6789' | 80.00 | 17 | 2 | '45.00EU/10.00US/7.50UK' |
| 'Ivy Anderson' | '555-012-3456' | 60.00 | 36 | 4 | '44.00EU/9.50US/7.00UK' |
| 'Larry Lee' | '555-345-6782' | 115.00 | 10 | 1 | '36.00EU/5.50US/3.00UK' |
| … | … | … | … | … | … |

# Section 4. Programmability (20 pts)

## Find All Orders By Email Address

Create a user-defined function, named **udf\_OrdersByEmail(@email)** that receives a user's email address, and returns the number of orders that the user has in the database.

### Example 1

|  |
| --- |
| **Query** |
| **SELECT dbo.udf\_OrdersByEmail('sstewart@example.com')** |
| **Output** |
| **3** |

### Example 2

|  |
| --- |
| **Query** |
| **SELECT dbo.udf\_OrdersByEmail('ohernandez@example.com')** |
| **Output** |
| **2** |

### Example 3

|  |
| --- |
| **Query** |
| **SELECT dbo.udf\_OrdersByEmail('nonexistent@example.com')** |
| **Output** |
| **0** |

## Shoe Details By Size

Your task is to create a **stored procedure** named **usp\_SearchByShoeSize**, which accepts the following parameters:

* **shoeSize(of type decimal with precision(5,2))**

Extract full information about the **orders** for **shoes** with the specified **EU size**. The needed data includes the **shoe model name**, **full name of the user** who ordered it, **level of the shoe price** (categorized as **'Low' – the price is lower than 100**, **'Medium' – the price is between 100 and 200**, or **'High' – the price is higher than 200**), **brand name**, and the **size of the shoe** (**EU size**).

The results should be ordered by **brand name**, and then by the **user's full name**.

Required columns:

* **ModelName**
* **UserFullName**
* **PriceLevel**
* **BrandName**
* **SizeEU**

### Example

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Query** | | | | |
| **EXEC usp\_SearchByShoeSize 40.00** | | | | |
| **Output** | | | | |
| **ModelName** | **UserFullName** | **PriceLevel** | **BrandName** | **SizeEU** |
| Blazer Mid | Nina Martin | Medium | Nike | 40.00 |
| Air Force 1 | Rachel Wood | Medium | Nike | 40.00 |
| ZoomX Vaporfly | Sam Stewart | High | Nike | 40.00 |
| … | … | … | … | … |