# Database Basics MS SQL Exam – 13 October 2019

Exam problems for the [“Database Basics” course @ SoftUni](https://softuni.bg/courses/databases-basics-ms-sql-server).

Submit your solutions in the SoftUni Judge system at <https://judge.softuni.bg/>

# Bitbucket

You’ve most likely heard of Github. Well … There is a side project called “Bitbucket” which is the back-up data of Github. You are one of the few selected to work in the multi-billion company, as one of the back-up database managers. You’ll need to prove your skills by designing and manipulating data in the Instagraph prototype.

# Section 1. DDL (30 pts)

You are given an E/R Diagram of the Trip Service:



Crеate a database called Bitbucket. You need to create **6 tables**:

* Users – contains information about the **users**.
* Repositories – contains information about the **repositories**.
* RepositoriesContributors – a **many** to **many** **mapping** table between the **repositories** and the **users**.
* Issues – contains information about the **issues**.
  + Each issue has a repository.
  + Each issue has an assignee (user).
* Commits – contains information about the **commits**.
  + Each commit **MAY** have an issue.
  + Each commit has a repository.
  + Each commit has a contributor (user).
* Files – contains information about the files.
  + Each file MAY have a parent (file).
  + Each file has a commit.

**Users**

|  |  |  |
| --- | --- | --- |
| **Column Name** | **Data Type** | **Constraints** |
| Id | **Integer** from **0** to **2,147,483,647** | Unique table **identificator**, **Identity** |
| Username | **String** up to 30 symbols | **NULL** is **not** allowed |
| Password | **String** up to 30 symbols | **NULL** is **not** allowed |
| Email | **String** up to 50 symbols | **NULL** is **not** allowed |

**Repositories**

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

**RepositoriesContributors**

|  |  |  |
| --- | --- | --- |
| **Column Name** | **Data Type** | **Constraints** |
| RepositoryId | **Integer** from **0** to **2,147,483,647** | **NULL** is **not** allowed, Relationship with table Repositories |
| ContributorId | **Integer** from **0** to **2,147,483,647** | **NULL** is **not** allowed, Relationship with table Users |

**Issues**

|  |  |  |
| --- | --- | --- |
| **Column Name** | **Data Type** | **Constraints** |
| Id | **Integer** from **0** to **2,147,483,647** | Unique table **identificator**, **Identity** |
| Title | **String** up to 255 symbols | **NULL** is **not** allowed |
| IssueStatus | **String** with **exactly 6** symbols | **NULL** is **not** allowed |
| RepositoryId | **Integer** from **0** to **2,147,483,647** | **NULL** is **not** allowed, Relationship with table Repositories |
| AssigneeId | **Integer** from **0** to **2,147,483,647** | **NULL** is **not** allowed, Relationship with table Users |

**Commits**

|  |  |  |
| --- | --- | --- |
| **Column Name** | **Data Type** | **Constraints** |
| Id | **Integer** from **0** to **2,147,483,647** | Unique table **identificator**, **Identity** |
| Message | **String** up to 255 symbols | **NULL** is **not** allowed |
| IssueId | **Integer** from **0** to **2,147,483,647** | Relationship with table Issues |
| RepositoryId | **Integer** from **0** to **2,147,483,647** | **NULL** is **not** allowed, Relationship with table Repositories |
| ContributorId | **Integer** from **0** to **2,147,483,647** | **NULL** is **not** allowed, Relationship with table Users |

**Files**

|  |  |  |
| --- | --- | --- |
| **Column Name** | **Data Type** | **Constraints** |
| Id | **Integer** from **0** to **2,147,483,647** | Unique table **identificator**, **Identity** |
| Name | **String** up to 100 symbols | **NULL** is **not** allowed |
| Size | **Decimal** number with **two-digit** precision | **NULL** is **not** allowed |
| ParentId | **Integer** from **0** to **2,147,483,647** | Relationship with table Files |
| CommitId | **Integer** from **0** to **2,147,483,647** | **NULL** is **not** allowed, Relationship with table Commits |

## Database Design

Submit all of yours **create** **statements** to Judge (only creation of tables).

CREATE TABLE Users(

[Id] INT PRIMARY KEY IDENTITY,

[Username] NVARCHAR(30) NOT NULL,

[Password] NVARCHAR(30) NOT NULL,

[Email] NVARCHAR(50) NOT NUll

)

CREATE TABLE Repositories(

[Id] INT PRIMARY KEY IDENTITY,

[Name] NVARCHAR(50) NOT NULL

)

CREATE TABLE RepositoriesContributors(

[RepositoryId] INT FOREIGN KEY REFERENCES Repositories(Id) NOT NULL,

[ContributorId] INT FOREIGN KEY REFERENCES Users(Id) NOT NULL

)

ALTER TABLE RepositoriesContributors

ADD CONSTRAINT PK\_RepositoryIdContributorId

PRIMARY KEY([RepositoryId], [ContributorId])

CREATE TABLE Issues(

[Id] INT PRIMARY KEY IDENTITY,

[Title] NVARCHAR(255) NOT NULL,

[IssueStatus] CHAR(6) NOT NULL,

[RepositoryId] INT FOREIGN KEY REFERENCES Repositories(Id) NOT NULL,

[AssigneeId] INT FOREIGN KEY REFERENCES Users(Id) NOT NULL

)

CREATE TABLE Commits(

[Id] INT PRIMARY KEY IDENTITY,

[Message] VARCHAR(255) NOT NULL,

[IssueId] INT FOREIGN KEY REFERENCES Issues(Id),

[RepositoryId] INT FOREIGN KEY REFERENCES Repositories(Id) NOT NULL,

[ContributorId] INT FOREIGN KEY REFERENCES Users(Id) NOT NULL

)

CREATE TABLE Files(

[Id] INT PRIMARY KEY IDENTITY,

[Name] VARCHAR(100) NOT NULL,

[Size] DECIMAL(18,2) NOT NULL,

[ParentId] INT FOREIGN KEY REFERENCES Files(Id),

[CommitId] INT FOREIGN KEY REFERENCES Commits(Id) NOT NULL

)

# Section 2. DML (10 pts)

**Before you start, you must import “**DataSet-Bitbucket.sql**”. If you have created the structure correctly, the data should be successfully inserted without any errors.**

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

## Insert

**Insert** some sample data into the database. Write a query to add the following records into the corresponding tables. **All Ids should be auto-generated**.

INSERT INTO Files

([Name], [Size], [ParentId], [CommitId])

VALUES

('Trade.idk', 2598.0, 1, 1),

('menu.net', 9238.31, 2, 2),

('Administrate.soshy', 1246.93, 3, 3),

('Controller.php', 7353.15, 4, 4),

('Find.java', 9957.86, 5, 5),

('Controller.json', 14034.87, 3, 6),

('Operate.xix', 7662.92, 7, 7)

INSERT INTO Issues([Title], [IssueStatus], [RepositoryId], [AssigneeId])

VALUES

('Critical Problem with HomeController.cs file','open', 1, 4),

('Typo fix in Judge.html','open', 4, 3),

('Implement documentation for UsersService.cs','closed', 8, 2),

('Unreachable code in Index.cs','open', 9, 8)

**Files**

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Size | ParentId | CommitId |
| Trade.idk | 2598.0 | 1 | 1 |
| menu.net | 9238.31 | 2 | 2 |
| Administrate.soshy | 1246.93 | 3 | 3 |
| Controller.php | 7353.15 | 4 | 4 |
| Find.java | 9957.86 | 5 | 5 |
| Controller.json | 14034.87 | 3 | 6 |
| Operate.xix | 7662.92 | 7 | 7 |

**Issues**

|  |  |  |  |
| --- | --- | --- | --- |
| Title | IssueStatus | RepositoryId | AssigneeId |
| Critical Problem with HomeController.cs file | open | 1 | 4 |
| Typo fix in Judge.html | open | 4 | 3 |
| Implement documentation for UsersService.cs | closed | 8 | 2 |
| Unreachable code in Index.cs | open | 9 | 8 |

## Update

Make issue status 'closed' where Assignee Id is 6.

UPDATE Issues

SET IssueStatus = 'closed'

WHERE AssigneeId = 6

## Delete

Delete repository "**Softuni-Teamwork**" in repository **contributors** and **issues**.

DELETE

FROM Issues

WHERE RepositoryId IN

(

SELECT Id

FROM Repositories

WHERE [Name] LIKE 'Softuni-Teamwork'

)

DELETE

FROM RepositoriesContributors

WHERE RepositoryId IN

(

SELECT Id

FROM Repositories

WHERE [Name] LIKE 'Softuni-Teamwork'

)

# Section 3. Querying (40 pts)

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

## Commits

Select all commits from the database. Order them by **id** (ascending), **message** (ascending), **repository id** (ascending) and **contributor id** (ascending).

SELECT [Id], [Message], [RepositoryId], [ContributorId]

FROM Commits

ORDER BY [Id] ASC,

[Message] ASC,

[RepositoryId] ASC,

[ContributorId] ASC

### Examples

|  |  |  |  |
| --- | --- | --- | --- |
| **Id** | **Message** | **RepositoryId** | **ContributorId** |
| 1 | Deleted deprecated functionality from index.cpp | 17 | 8 |
| 2 | Created README.MD | 14 | 8 |
| 3 | Initial Commit | 24 | 1 |
| 4 | Implemented config.json functionality | 10 | 12 |
| … | … | … | … |

## Heavy HTML

Select all of the **files**, which have **size, greater** than **1000**, and their **name** contains "html". Order them by **size** (descending), **id** (ascending) and by **file name** (ascending)

SELECT Id, Name, Size

FROM Files

WHERE Size > 1000 AND [Name] LIKE '%html%'

ORDER BY Size DESC, Id ASC, [Name] ASC

### Examples

|  |  |  |
| --- | --- | --- |
| **Id** | **Name** | **Size** |
| 49 | compile.html | 27402.59 |
| 17 | Login.html | 2863.23 |
| … | … | .. |

## Issues and Users

Select all of the issues, and the users that are **assigned** to them, so that they end up in the following format: {**username**} : {**issueTitle**}. Order them by **issue id** (descending) and **issue assignee** (ascending).

SELECT i.Id AS [Id], CONCAT(u.Username, ' : ', i.Title) AS [IssueAssignee]

FROM Issues as i

JOIN Users as u

ON i.AssigneeId = u.Id

ORDER BY i.Id DESC, [IssueAssignee]

### Examples

|  |  |
| --- | --- |
| **Id** | **IssueAssignee** |
| 75 | TheDivineBel : Critical bug in Controller.php ruins application when executed |
| 74 | DarkImmagidsa : Compilation failed while trying to execute init.xml |
| 73 | ScoreAntigarein : Loose Cohesion and Strong Coupling in Beat.html |
| … | … |

## Non-Directory Files

### Examples

Select all of the **files**, which are NOT a **parent** to any other file. Select their size of the file and add "**KB**" to the end of it. Order them file **id** (ascending), **file name** (ascending) and **file size** (descending).

SELECT d.Id, d.Name, CONCAT(d.Size,'KB') AS [Size]

FROM Files AS f

RIGHT JOIN Files AS d

ON f.ParentId = d.Id

WHERE f.ParentId IS NULL

ORDER BY d.Id ASC, d.Name ASC, [Size] DESC

|  |  |  |
| --- | --- | --- |
| **Id** | **Name** | **Size** |
| 6 | Controller.json | 14034.87KB |
| 12 | Model.MD | 4753.67KB |
| 17 | Login.html | 2863.23KB |
| … | … | .. |

## Most Contributed Repositories

Select the **top 5** repositories in terms of **count** of **commits**. Order them by **commits count** (descending), **repository id** (ascending) then by **repository name** (ascending).

SELECT TOP(5) r.Id, r.Name, COUNT(c.Id) AS [Commits]

FROM Repositories AS r

JOIN RepositoriesContributors AS rc

ON r.Id = rc.RepositoryId

JOIN Users AS u

ON rc.ContributorId = u.Id

JOIN Commits AS c

ON r.Id = c.RepositoryId

GROUP BY r.Id, r.Name

ORDER BY COUNT(c.Id) DESC, r.Id ASC, r.Name

### Examples

|  |  |  |
| --- | --- | --- |
| **Id** | **Name** | **Commits** |
| 1 | WorkWork | 20 |
| 7 | DundaApp | 16 |
| 10 | SortedTupleJS | 12 |
| … | … | .. |

## User and Files

Select all users which have **commits**. Select their username and average size of the file, which were uploaded by them. Order the results by **average upload size** (descending) and by **username** (ascending).

SELECT u.Username, AVG(f.Size) AS Size

FROM Users AS u

JOIN Commits AS c

ON u.Id = c.ContributorId

JOIN Files AS f

ON c.Id = f.CommitId

GROUP BY u.Username

ORDER BY AVG(f.Size) DESC, u.Username

### Examples

|  |  |
| --- | --- |
| **Username** | **Size** |
| RoundInspecindi | 19506.877500 |
| BlaAntigadsa | 18397.434000 |
| … | … |

# Section 4. Programmability (20 pts)

## User Total Commits

Create a **user defined function**, named **udf\_UserTotalCommits(@username)** that receives a username.

The function must return count of all commits for the user:

CREATE FUNCTION udf\_UserTotalCommits(@username VARCHAR(MAX))

RETURNS INT

--The function must return count of all commits for the user:

AS

BEGIN

RETURN(SELECT COUNT(\*)

FROM Users AS u

JOIN Commits AS c

ON u.Id = c.ContributorId

WHERE u.Username = @username)

END

### Example:

|  |
| --- |
| **Query** |
| **SELECT dbo.udf\_UserTotalCommits('UnderSinduxrein')** |
| **Output** |
| **6** |

## Find by Extensions

Create a **user defined stored procedure**, named **usp\_FindByExtension(@extension)**, that receives a files extensions.  
The procedure must print the **id**, **name** and **size** of the file. Add "**KB**" in the end of the size. Order them by **id** (ascending), **file name** (ascending) and **file size** (descending)

CREATE PROCEDURE usp\_FindByExtension(@extension VARCHAR(MAX))

AS

BEGIN

SELECT Id, Name, CONCAT(Size,'KB') AS [Size]

FROM Files

WHERE Name LIKE '%' + @extension

ORDER BY Id ASC, Name ASC, Size Desc

END

### Example:

|  |  |  |
| --- | --- | --- |
| **Query** | | |
| **EXEC** **usp\_FindByExtension** **'txt'** | | |
| **Output** | | |
| **Id** | **Name** | **Size** |
| 28 | Jason.txt | 10317.54KB |
| 31 | file.txt | 5514.02KB |
| 43 | init.txt | 16089.79KB |