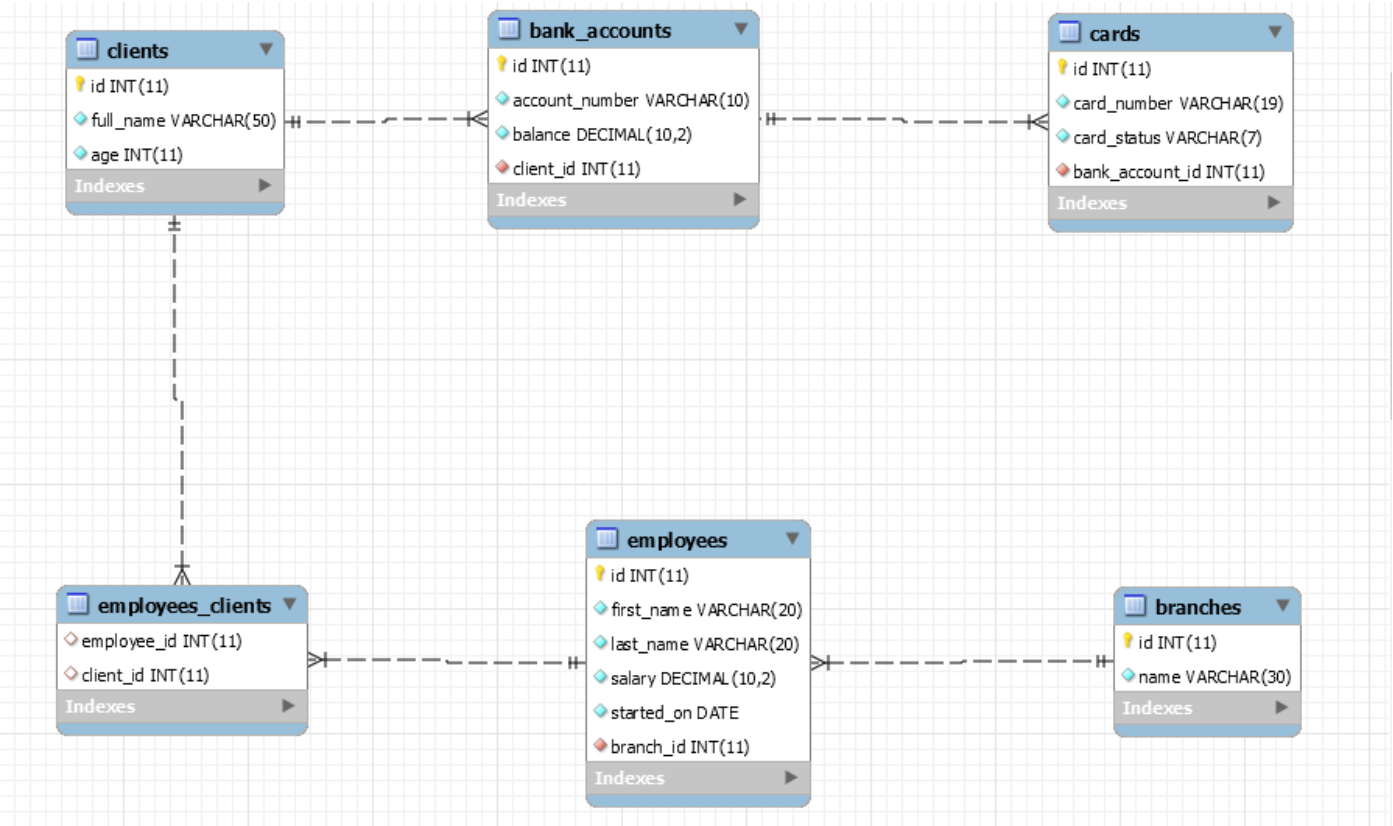
MySQL Exam

Royal United Kingsman – Bank

Royal United Kingsman Bank or most widely known as R.U.K. Bank is a new bank founded by Darkman Nakov. You have been employed by the bank to design a database prototype, which will lay the foundation for the main database. You will need to prove your skills in database definition, data manipulation and extraction and database programmability.

0. Section 0: Database Overview

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



The **Bank's Database** needs to hold information about **branches**, **employees**, **clients**, **bank accounts**, **cards**.

Your task is to create a database called **`ruk_database`**. Then you will have to create several **tables**.

- **`branches`** – contains information about the **branches**.
- **`employees`** – contains information about the **employees**.
  - Each **`employee`** has a **`branch`**.
- **`clients`** – contains information about the **clients**.
- **`employees_clients`** – a **many** to **many mapping** table between the **employees** and the **clients**.
- **`bank_accounts`** – contains information about the **bank accounts**.
  - Each **`bank_account`** has a **`client`**.

- **`cards`** – contains information about the **cards**.
  - Each **`card`** has a **`client`**.
  - Each **`card`** has a **`bank_account`**.

SoftUni Foundation
© Software University Foundation. This work is licensed under the CC-BY-NC-SA license.
Follow us:

1. 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.

01. Table Design

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

branches

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 **30 characters**. Unicode is **NOT** needed.	**NULL** is **NOT** permitted. **UNIQUE** values.

employees

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.
`salary`	**DECIMAL**, up to **10 digits**, **2** of which after the **decimal point**.	**NULL** is **NOT** permitted.
`started_on`	A **DATE** field. Format - (**YYYY-MM-DD**).	**NULL** is **NOT** permitted.
`branch_id`	**Integer,** from **1** to **2,147,483,647.**	Relationship with table `branches`. **NULL** is **NOT** permitted.

clients

Column Name	Data Type	Constraints
`id`	**Integer,** from **1** to **2,147,483,647.**	**Primary Key** **AUTO_INCREMENT**
`full_name`	A **string** containing a maximum of **50 characters**. Unicode is **NOT** needed.	**NULL** is **NOT** permitted.
`age`	**Integer,** from **1** to **2,147,483,647.**	**NULL** is **NOT** permitted.

employees_clients

© Software University Foundation. This work is licensed under the CC-BY-NC-SA license.

Column Name	Data Type	Constraints
`employee_id`	**Integer**, from **1** to **2,147,483,647.**	Relationship with table **`employees`**.
`client_id`	**Integer**, from **1** to **2,147,483,647.**	Relationship with table **`clients`**.

bank_accounts

Column Name	Data Type	Constraints
`id`	**Integer**, from **1** to **2,147,483,647.**	**Primary Key** **AUTO_INCREMENT**
`account_number`	A **string** containing a maximum of **10 characters**. Unicode is **NOT** needed.	**NULL** is **NOT** permitted.
`balance`	**DECIMAL**, up to **10 digits**, **2** of which after the **decimal point**.	**NULL** is **NOT** permitted**.**
`client_id`	**Integer**, from **1** to **2,147,483,647.**	Relationship with table **`clients`**. **NULL** is **NOT** permitted**.** **UNIQUE** values.

cards

Column Name	Data Type	Constraints
`id`	**Integer**, from **1** to **2,147,483,647.**	**Primary Key** **AUTO_INCREMENT**
`card_number`	A **string** containing a maximum of **19 characters**. Unicode is **NOT** needed.	**NULL** is **NOT** permitted.
`card_status`	A **string** containing a maximum of **7 characters**. Unicode is **NOT** needed.	**NULL** is **NOT** permitted.
`bank_account_id`	**Integer**, from **1** to **2,147,483,647.**	Relationship with table **`bank_accounts`**. **NULL** is **NOT** permitted**.**

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.

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

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

02. Insert

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

For **`clients`** with **`id`** between **191** and **200** (**inclusive**), **insert data** in the **`cards`** table with the **following values**:

© Software University Foundation. This work is licensed under the CC-BY-NC-SA license.

- **`card_number`** – set it to **full name** of the **client**, but **reversed**!
- **`card_status`** – set it to "**`Active`**".
- **`bank_account_id`** –set it to **client's id** value.

03. Update

Update all **clients** which have the same **id** as the **employee** they are appointed to. Set their **employee_id** with the **employee** with the **lowest count** of **clients**.

If there are 2 such **employees** with equal **count** of **clients**, take the one with the **lowest id**.

04. Delete

R.U.K. Bank is a sophisticated network. As such, it cannot allow procrastination and lazy behavior.

Delete all **`employees`** which do not have any clients.

3. 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.

05. Clients

Extract from the database, all of the **clients**.

Order the results ascending by **client id**.

Required Columns

- **`id (clients)`**
- **`full_name`**

Example

id	full_name
1	Hunter Wesgate
...	...

06. Newbies

One of your bosses has requested a functionality which checks the newly employed – highly paid people.

Extract from the database, all of the **employees**, which have **salary** greater than or equal to **100000** and have started **later** than or **equal** to the 1st of January - 2018.

The **salary** should have a "**$**" as a **prefix**.

Order the results **descending** by **`salary`**, then by **id**.

Required Columns

- **`id (employees)`**

© Software University Foundation. This work is licensed under the CC-BY-NC-SA license.

- `full_name (first_name + " " + last_name)`
- `salary`
- `started_on`

Example

id	full_name	salary	started_on
41	Lisbeth Skett	$981421.79	2018-04-16
...	...	...	

07. Cards against Humanity

Extract from the database, all of the **cards**, and the **clients** that own them, so that they end up in the following format:

`{card_number} : {full_name}`

Order the results **descending** by **card id**.

Required Columns

- `id (cards)`
- `card_token`

Example

id	card_token
500	SM80 M775 4918 653X : Erin Cullingworth
...	...

08. Top 5 Employees

Extract from the database, the top 5 **employees**, in terms of **clients** assigned to them.

Order the results descending by **count of clients**, and ascending by **employee id**.

Required Columns

- `name (employees)`
- `started_on`
- `count_of_clients`

Example

name	started_on	count_of_clients
Trula Glasscott	2017-08-23	14
...	...	...

SoftUni Foundation
© Software University Foundation. This work is licensed under the CC-BY-NC-SA license.
Follow us:

09. Branch cards

Extract from the database, **all branches** with the count of their issued cards. Order the results by the **count of cards,** then by **branch name**.

Required Columns

- `name (branch)`
- `count_of_cards`

Example

name	count_of_cards
Becker Branch	93
Mifflin Branch	82
Mendota Branch	67
Moulton Branch	58

4. 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.

10. Extract client cards count

Create a **user defined function** with the name **`udf_client_cards_count(name VARCHAR(30))`** that receives a **client's full name** and returns the number of cards he has.

Required Columns

- `full_name (clients)`
- `cards (count of cards)`

Example

Query

```
SELECT c.full_name, udf_count_of_cards('Baxy David') as `cards` FROM clients c
WHERE c.full_name = 'Baxy David';
```

full_name	cards
Baxy David	6

11. Extract Client Info

Create a stored procedure **`udp_clientinfo`** which accepts the following parameters:

- `full_name`

And extracts data about the **client** with the given **full name**.

© Software University Foundation. This work is licensed under the CC-BY-NC-SA license.

Aside from the **`full_name`**, the procedure should extract the **client**'s **age**, **bank account number** and **balance**.

The **account**'s **salary** should have "**$**" prefix.

```
CALL udp_clientinfo('Hunter Wesgate');
```

Result

full_name	age	account_number	balance
Hunter Wesgate	33	69666616-8	$803355.32

© Software University Foundation. This work is licensed under the CC-BY-NC-SA license.