**Задание семинара 9**

**Лим В. БПИ225**

**1. Maths with String Manipulations**

Given a demographics table in the following format:

demographics table schema

* id
* name
* birthday
* race

return a single column named calculation where the value is the bit length of name, added to the number of characters in race.

**Решение:**

SELECT

bit\_length(name) + char\_length(race) AS calculation

FROM

demographics;

**2. Bit Length**

Given a demographics table in the following format:

\*\* demographics table schema \*\*

* id
* name
* birthday
* race

you need to return the same table where all text fields (name & race) are changed to the bit length of the string.

**Решение:**

SELECT

id,

bit\_length(name) AS name,

birthday,

bit\_length(race) AS race

FROM

demographics;

**3. ASCII Converter**

Given a demographics table in the following format:

\*\* demographics table schema \*\*

* id
* name
* birthday
* race

you need to return the same table where all text fields (name & race) are changed to the ascii code of their first byte.

e.g. Verlie = 86 Warren = 87 Horace = 72 Tracy = 84

**Решение:**

SELECT

id,

ascii(substr(name, 1, 1)) AS name,

birthday,

ascii(substr(race, 1, 1)) AS race

FROM

demographics;

**4. Concatenating Columns**

Given the table below:

\*\* names table schema \*\*

* id
* prefix
* first
* last
* suffix

Your task is to use a select statement to return a single column table containing the full title of the person (concatenate all columns together except id), as follows:

\*\* output table schema \*\*

* title

Don't forget to add spaces.

**Решение:**

SELECT

TRIM(CONCAT\_WS(' ', prefix, first, last, suffix)) AS title

FROM

names;

**5. Padding Encryption**

You are given a table with the following format:

\*\* encryption table schema \*\*

* md5
* sha1
* sha256

Problem is the table looks so unbalanced

* the sha256 column contains much longer strings. You need to balance things up. Add '1' to the end of the md5 addresses as many times as you need to to make them the same length as those in the sha256 column. Add '0' to the beginning of the sha1 values to achieve the same result.  
  Return as:  
  \*\* output table schema \*\*
* md5
* sha1
* sha256

**Решение:**

SELECT

md5 || repeat('1', char\_length(sha256) - char\_length(md5)) AS md5,

repeat('0', char\_length(sha256) - char\_length(sha1)) || sha1 AS sha1,

sha256

FROM

encryption;

**6. Right and Left**

You are given a table named repositories, format as below:

\*\* repositories table schema \*\*

* project
* commits
* contributors
* address

The table shows project names of major cryptocurrencies, their numbers of commits and contributors and also a random donation address ( not linked in any way :) ).

For each row: Return first x characters of the project name where x = commits. Return last y characters of each address where y = contributors.

Return project and address columns only, as follows:

\*\* output table schema \*\*

* project
* address

Case should be maintained.

**Решение:**

SELECT

LEFT(project, commits) AS project,

RIGHT(address, contributors) AS address

FROM

repositories;

**7. Regex Replace**

You are given a table named repositories, format as below:

\*\* repositories table schema \*\*

* project
* commits
* contributors
* address

The table shows project names of major cryptocurrencies, their numbers of commits and contributors and also a random donation address ( not linked in any way :) ).

Your job is to remove all numbers in the address column and replace with '!', then return a table in the following format:

\*\* output table schema \*\*

* project
* commits
* contributors
* address

Case should be maintained.

**Решение:**

SELECT

project,

commits,

contributors,

regexp\_replace(address, '\d', '!', 'g') AS address

FROM

repositories;

**8. Real Price!**

You are the owner of the Grocery Store. All your products are in the database :)

Customer often need to now how much really they pay for the products. Manufacturers make different sizes of same product so it is hard to compare prices, sometimes they make packages look big, but the weight of the product is small.

Make a SELECT query which will tell the price per kg of the product.

Weight is in grams! Round the price\_per\_kg to 2 decimal places.

Order results by price\_per\_kg ascending, then by name ascending.

Products table schema:

* id (int)
* name (string)
* price (float)
* stock (int)
* weight (float)
* producer (string)
* country (string)

Results table schema:

* name (string)
* weight (float)
* price (float)
* price\_per\_kg (float)

**Решение:**

SELECT

name,

weight,

price,

ROUND((price / (weight / 1000)), 2) AS price\_per\_kg

FROM

products

ORDER BY

price\_per\_kg ASC,

name ASC;