# *Отчет к ДЗ Взаимодействия со словарями*

Задачи:

1. Создать таблицу с полями:
   user_id UInt64,
   action String,
   expense UInt64
2. Создать словарь, в качестве ключа user_id, в качестве атрибута email String, источник словаря любой вам удобный, например file.
3. Наполнить таблицу и источник любыми данными, с низкоардинальными значениями для поля action и хотя бы по несколько повторящихся строк для каждого user_id
4. написать SELECT, возвращающий:
   - email при помощи dictGet,
   - аккамулятивную сумму expense, с окном по action
   - сортировка по email

1. Создаем таблицу

```
NB-R911LJCT.o3.ru :) CREATE TABLE user_actions (
    user_id UInt64,
    action String,
    expense UInt64
) ENGINE = MergeTree()
ORDER BY user_id;

CREATE TABLE user_actions
(
    `user_id` UInt64,
    `action` String,
    `expense` UInt64
)
ENGINE = MergeTree
ORDER BY user_id

Query id: 81eabbe4-d4f1-4e4c-94ca-97b2bf0a830f

Ok.

0 rows in set. Elapsed: 0.022 sec.

NB-R911LJCT.o3.ru :)
```

2. Создать словарь из файла csv.

```
NB-R911LJCT.o3.ru :) CREATE DICTIONARY user_emails
(
    user_id UInt64,
    email String
)
PRIMARY KEY user_id
SOURCE(file(PATH 'var/lib/clickhouse/user_files/users.csv' FORMAT 'CSVWithNames'))
LIFETIME(MIN 1 MAX 1)
LAYOUT(flat);

CREATE DICTIONARY user_emails
(
    `user_id` UInt64,
    `email` String
)
PRIMARY KEY user_id
SOURCE(FILE(PATH 'var/lib/clickhouse/user_files/users.csv' FORMAT 'CSVWithNames'))
LIFETIME(MIN 1 MAX 1)
LAYOUT(FLAT)

Query id: ec780c89-b0a6-4c1a-81a4-b573b9cc468d

Ok.

0 rows in set. Elapsed: 0.014 sec.
```

```
NB-R911LJCT.o3.ru :) SELECT * FROM system.dictionaries Format Vertical

SELECT *
FROM system.dictionaries
FORMAT Vertical

Query id: 5482bcaf-46ce-4e41-8dc0-5a037f3851f9

Row 1:
──────
database:                           default
name:                               user_emails
uuid:                               2bec4b08-dbfe-4a48-92b6-18bb85f702fd
status:                             NOT_LOADED
origin:                             2bec4b08-dbfe-4a48-92b6-18bb85f702fd
type:
key.names:                          ['user_id']
key.types:                          ['UInt64']
attribute.names:                    ['email']
attribute.types:                    ['String']
bytes_allocated:                    0
hierarchical_index_bytes_allocated: 0
query_count:                        0
hit_rate:                           0
found_rate:                         0
element_count:                      0
load_factor:                        0
source:
lifetime_min:                       0
lifetime_max:                       0
loading_start_time:                 1970-01-01 03:00:00
last_successful_update_time:        1970-01-01 03:00:00
loading_duration:                   0
last_exception:
comment:

1 row in set. Elapsed: 0.002 sec.
```

3. Наполнить таблицу и источник любыми данными

```
NB-R911LJCT.o3.ru :) INSERT INTO user_actions (user_id, action, expense) VALUES
(1, 'login', 100),
(1, 'logout', 50),
(1, 'login', 150),
(2, 'login', 200),
(2, 'logout', 100),
(2, 'login', 250),
(3, 'login', 300),
(3, 'logout', 150);
```

4. SELECT, возвращающий:
   - email при помощи dictGet,
   - аккамулятивную сумму expense, с окном по action
   - сортировка по email

```
NB-R911LJCT.o3.ru :) SELECT
    dictGet('user_emails', 'email', user_id) AS email,
    action,
    sum(expense) OVER (PARTITION BY action ORDER BY user_id ROWS BETWEEN UNBOUNDED PRECEDING AND CURRENT ROW) AS cumulative_expense
FROM
    user_actions
ORDER BY
    email;


SELECT
    dictGet('user_emails', 'email', user_id) AS email,
    action,
    sum(expense) OVER (PARTITION BY action ORDER BY user_id ASC ROWS BETWEEN UNBOUNDED PRECEDING AND CURRENT ROW) AS cumulative_expense
FROM user_actions
ORDER BY email ASC

Query id: b7a97e49-cd0f-4288-88f2-fb9f0f447901

    ┌─email─────────────┬─action─┬─cumulative_expense─┐
1.  │ user1@example.com │ login  │                100 │
2.  │ user1@example.com │ login  │                250 │
3.  │ user1@example.com │ logout │                 50 │
4.  │ user2@example.com │ login  │                450 │
5.  │ user2@example.com │ login  │                700 │
6.  │ user2@example.com │ logout │                150 │
7.  │ user3@example.com │ login  │               1000 │
8.  │ user3@example.com │ logout │                300 │
    └───────────────────┴────────┴────────────────────┘

8 rows in set. Elapsed: 0.011 sec.
```