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

```
CREATE TABLE default.test_tbl
(
    `user_id` UInt64,
    `action` String,
    `expense` UInt64,
    `email` String

)
ENGINE = MergeTree
ORDER BY user_id
SETTINGS index_granularity = 8192
```

```
CREATE DICTIONARY dict_test
(
    `user_id` UInt64,
    `action` String,
    `expense` UInt64,
    `email` String
)
PRIMARY KEY user_id
SOURCE(CLICKHOUSE(db default table test_tbl))
LAYOUT(FLAT)
LIFETIME(100)
```

```
INSERT INTO test_tbl (user_id, action, expense, email) VALUES (1, 'Buy', 300, '1@otus.ru');
INSERT INTO test_tbl (user_id, action, expense, email) VALUES (2, 'Move', 500, '2@otus.ru');
INSERT INTO test_tbl (user_id, action, expense, email) VALUES (2, 'Insert', 700, '2@otus.ru');
INSERT INTO test_tbl (user_id, action, expense, email) VALUES (2, 'Select', 8700, '2@otus.ru');
INSERT INTO test_tbl (user_id, action, expense, email) VALUES (3, 'Eat', 300, '3@otus.ru');
INSERT INTO test_tbl (user_id, action, expense, email) VALUES (3, 'Sleep', 999,'3@otus.ru');
```

4.написать SELECT, возвращающий:

- email при помощи dictGet,
- аккамулятивную сумму expense, с окном по action
- сортировка по email

```
envy :)  SELECT dictGet('dict_test', 'email', 2)

SELECT dictGet('dict_test', 'email', 2)

Query id: 34a8dd9f-bce3-4182-90cf-b9ae97b5f864

    ┌─dictGet('dict_test', 'email', 2)─┐
1.  │ 2@otus.ru                        │
    └──────────────────────────────────┘

1 row in set. Elapsed: 0.002 sec.

envy :)
```

```
envy :) SELECT action,  sum(expense) OVER(PARTITION BY action) from test_tbl

SELECT
    action,
    sum(expense) OVER (PARTITION BY action)
FROM test_tbl

Query id: effbe003-cf16-44fa-a1b1-45552eeec2bf

    ┌─action─┬─sum(expense) OVER (PARTITION BY action)─┐
1.  │ Buy    │                                     300 │
2.  │ Eat    │                                     300 │
3.  │ Insert │                                     700 │
4.  │ Move   │                                     500 │
5.  │ Select │                                    8700 │
6.  │ Sleep  │                                     999 │
    └────────┴─────────────────────────────────────────┘

6 rows in set. Elapsed: 0.004 sec.

envy :)
```

```
envy :)  SELECT action, email,  sum(expense) OVER(PARTITION BY action ORDER BY email) from test_tbl

SELECT
    action,
    email,
    sum(expense) OVER (PARTITION BY action ORDER BY email ASC)
FROM test_tbl

Query id: 0e6fc441-6e8d-4d3a-953d-33d0a92e95fa

    ┌─action─┬─email─────┬─sum(expense) OVER (PARTITION BY action ORDER BY email ASC)─┐
1.  │ Buy    │ 1@otus.ru │                                                        300 │
2.  │ Eat    │ 3@otus.ru │                                                        300 │
3.  │ Insert │ 2@otus.ru │                                                        700 │
4.  │ Move   │ 2@otus.ru │                                                        500 │
5.  │ Select │ 2@otus.ru │                                                       8700 │
6.  │ Sleep  │ 3@otus.ru │                                                        999 │
    └────────┴───────────┴────────────────────────────────────────────────────────────┘

6 rows in set. Elapsed: 0.004 sec.

envy :)
```