

**122022010 Shrayank Mistry**

**Description:**

- 1) Implement hash based natural join
- 2) Implement sort based natural join

**Programming Language:** Python

*Hash based Natural Join*

*Input Relations A, B with hash function that returns ascii value*

```
1  from tabulate import tabulate
2
3  /* Hash function return ascii value
4  def hash_function(h):
5      return ord(str(h))
6
7  table_A = [
8      ['CustomerId', 'CustomerName'],
9      [1, 'C1'],
10     [2, 'C2'],
11     [3, 'C3']
12 ]
13
14 table_B = [
15     ['InvoiceId', 'CustomerId', 'Total'],
16     [1, 2, '$44.20'],
17     [2, 2, '$13.37'],
18     [3, 1, '$144.5'],
19     [4, 3, '$501.1'],
20     [5, 3, '$66.77'],
21     [6, 1, '$100.0']
22 ]
```

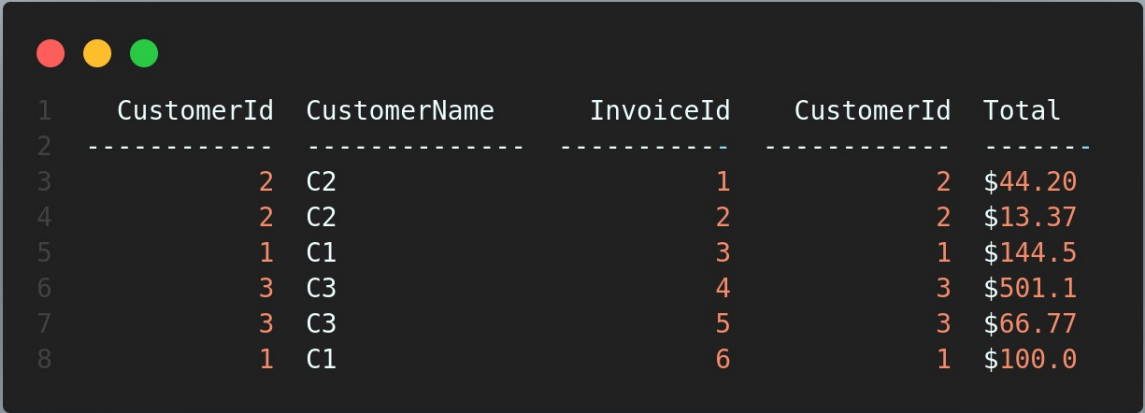
## 1) Build Phase

```
1  /* Build Phase
2  build_table = {}
3
4  for i in range(1, len(table_A)):
5      h = hash_function(table_A[i][0])
6      if build_table.get(h) == None:
7          build_table[h] = list()
8      t = [table_A[i][j] for j in range(0, len(table_A[i]))]
9      build_table[h].append(t)
```

## 2) Natural Join

```
1  joined_table = list()
2  /* Natural Join Phase
3
4  for i in range(1, len(table_B)):
5      h = hash_function(table_B[i][1])
6      if build_table.get(h) == None:
7          continue
8      records_A = build_table[h]
9      t = [table_B[i][j] for j in range(0, len(table_B[i]))]
10     for record in records_A:
11         joined_table.append(list(record + t))
12
13     # for record in joined_table:
14     #     print(record)
15
16     headers = table_A[0] + table_B[0]
17     print(tabulate(joined_table, headers))
```

3) Output



	CustomerId	CustomerName	InvoiceId	CustomerId	Total
1					
2					
3	2	C2	1	2	\$44.20
4	2	C2	2	2	\$13.37
5	1	C1	3	1	\$144.5
6	3	C3	4	3	\$501.1
7	3	C3	5	3	\$66.77
8	1	C1	6	1	\$100.0

## Sort Based Natural Join

Input Relations A, B with sorting function on CustomerId

```
1  from tabulate import tabulate
2
3  def sort_for_B(L):
4      return L[1]
5
6  def sort_for_A(L):
7      return L[0]
8
9  table_A = [
10     ['CustomerId', 'CustomerName'],
11     [1, 'C1'],
12     [2, 'C2'],
13     [3, 'C3']
14 ]
15
16 table_B = [
17     ['InvoiceId', 'CustomerId', 'Total'],
18     [1, 2, '$44.20'],
19     [2, 2, '$13.37'],
20     [3, 1, '$144.5'],
21     [4, 3, '$501.1'],
22     [5, 3, '$66.77'],
23     [6, 1, '$100.0']
24 ]
```

## 1) Sorting Relations A, B on CustomerId

```
1 headers = table_A[0] + table_B[0]
2
3 /* Sorting tables A, B on customerId
4 table_B = sorted(table_B[1:], key=sort_for_B)
5 table_A = sorted(table_A[1:], key=sort_for_A)
```

## 2) Natural Join

```
1 joined_table = list()
2 /* Natural Join Phase
3 j = 0
4 for i in range(len(table_A)):
5     while j < len(table_B):
6         if table_A[i][0] == table_B[j][1]:
7             joined_table.append(list(table_A[i] + table_B[j]))
8             j = j + 1
9         else:
10            break
11
12 print(tabulate(joined_table, headers))
```

## 3) Output

	CustomerId	CustomerName	InvoiceId	CustomerId	Total
	-----	-----	-----	-----	-----
3	1	C1	3	1	\$144.5
4	1	C1	6	1	\$100.0
5	2	C2	1	2	\$44.20
6	2	C2	2	2	\$13.37
7	3	C3	4	3	\$501.1
8	3	C3	5	3	\$66.77