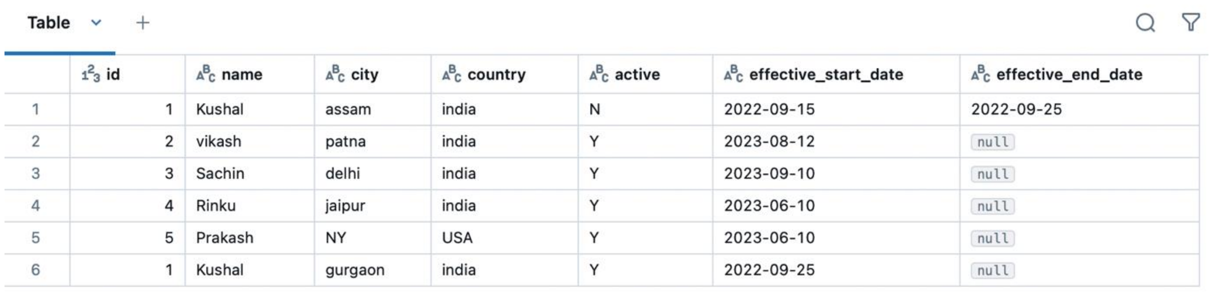
SCD TYPE 2 Implementation using PySpark

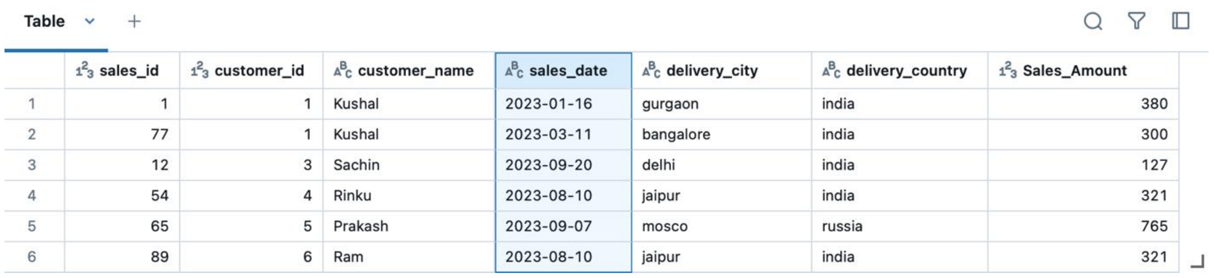
Customer Data:



Table

	id	name	city	country	active	effective_start_date	effective_end_date
1	1	Kushal	assam	india	N	2022-09-15	2022-09-25
2	2	vikash	patna	india	Y	2023-08-12	null
3	3	Sachin	delhi	india	Y	2023-09-10	null
4	4	Rinku	jaipur	india	Y	2023-06-10	null
5	5	Prakash	NY	USA	Y	2023-06-10	null
6	1	Kushal	gurgaon	india	Y	2022-09-25	null

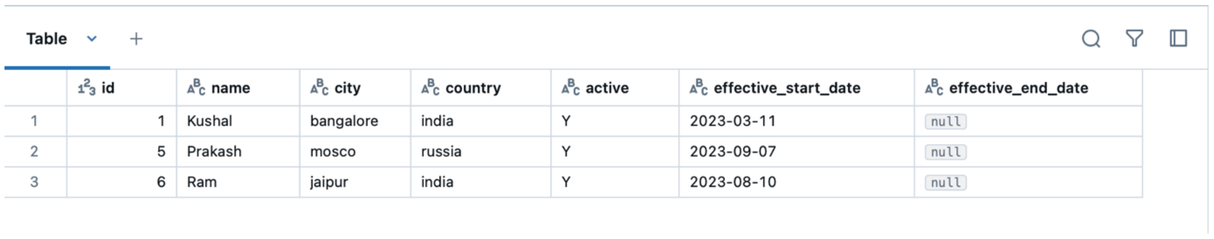
Sales Data:



Table

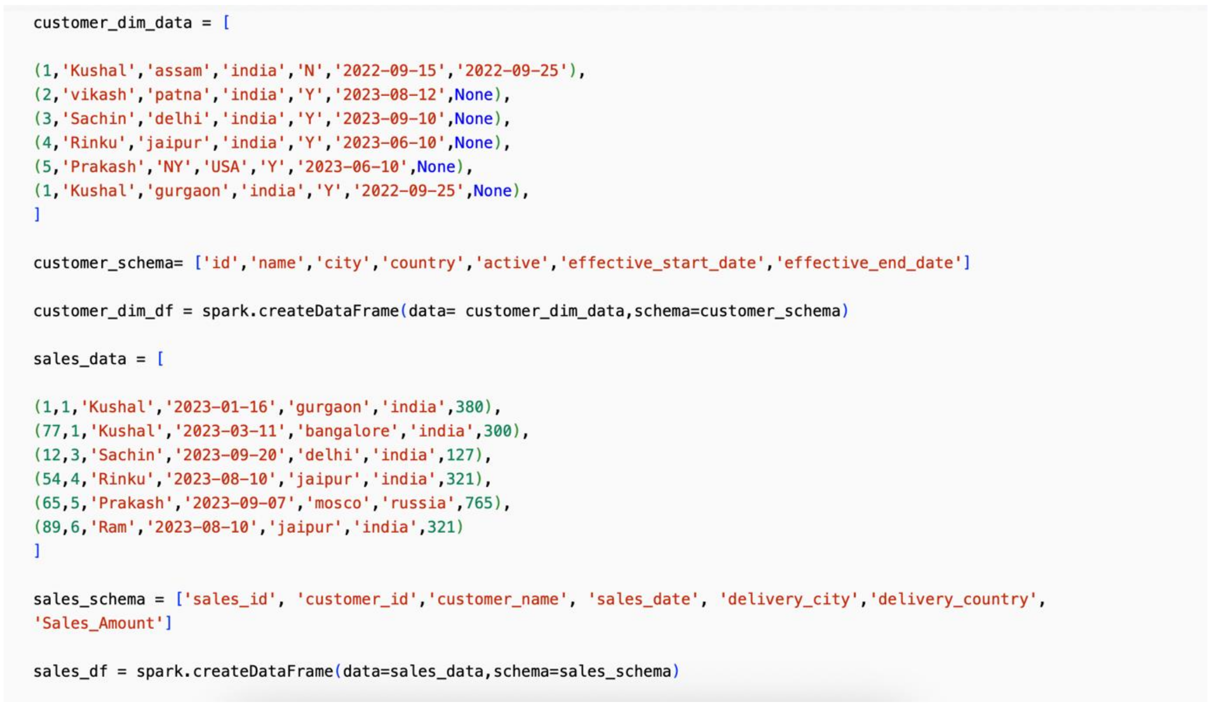
	sales_id	customer_id	customer_name	sales_date	delivery_city	delivery_country	Sales_Amount
1	1	1	Kushal	2023-01-16	gurgaon	india	380
2	77	1	Kushal	2023-03-11	bangalore	india	300
3	12	3	Sachin	2023-09-20	delhi	india	127
4	54	4	Rinku	2023-08-10	jaipur	india	321
5	65	5	Prakash	2023-09-07	mosco	russia	765
6	89	6	Ram	2023-08-10	jaipur	india	321

Expected-Output:

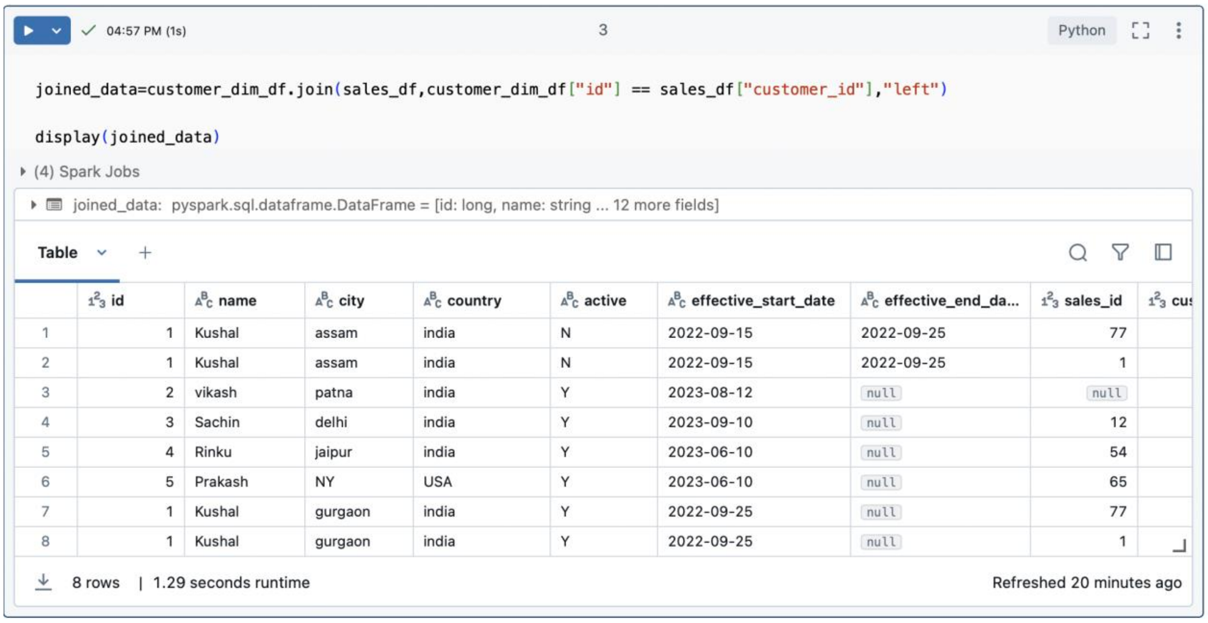


Table

	id	name	city	country	active	effective_start_date	effective_end_date
1	1	Kushal	bangalore	india	Y	2023-03-11	null
2	5	Prakash	mosco	russia	Y	2023-09-07	null
3	6	Ram	jaipur	india	Y	2023-08-10	null



```python
customer_dim_data = [

(1,'Kushal','assam','india','N','2022-09-15','2022-09-25'),
(2,'vikash','patna','india','Y','2023-08-12',None),
(3,'Sachin','delhi','india','Y','2023-09-10',None),
(4,'Rinku','jaipur','india','Y','2023-06-10',None),
(5,'Prakash','NY','USA','Y','2023-06-10',None),
(1,'Kushal','gurgaon','india','Y','2022-09-25',None),
]

customer_schema= ['id','name','city','country','active','effective_start_date','effective_end_date']

customer_dim_df = spark.createDataFrame(data= customer_dim_data,schema=customer_schema)

sales_data = [

(1,1,'Kushal','2023-01-16','gurgaon','india',380),
(77,1,'Kushal','2023-03-11','bangalore','india',300),
(12,3,'Sachin','2023-09-20','delhi','india',127),
(54,4,'Rinku','2023-08-10','jaipur','india',321),
(65,5,'Prakash','2023-09-07','mosco','russia',765),
(89,6,'Ram','2023-08-10','jaipur','india',321)
]

sales_schema = ['sales_id', 'customer_id','customer_name', 'sales_date', 'delivery_city','delivery_country',
'Sales_Amount']

sales_df = spark.createDataFrame(data=sales_data,schema=sales_schema)
```



```python
joined_data=customer_dim_df.join(sales_df,customer_dim_df["id"] == sales_df["customer_id"],"left")

display(joined_data)
```

(4) Spark Jobs

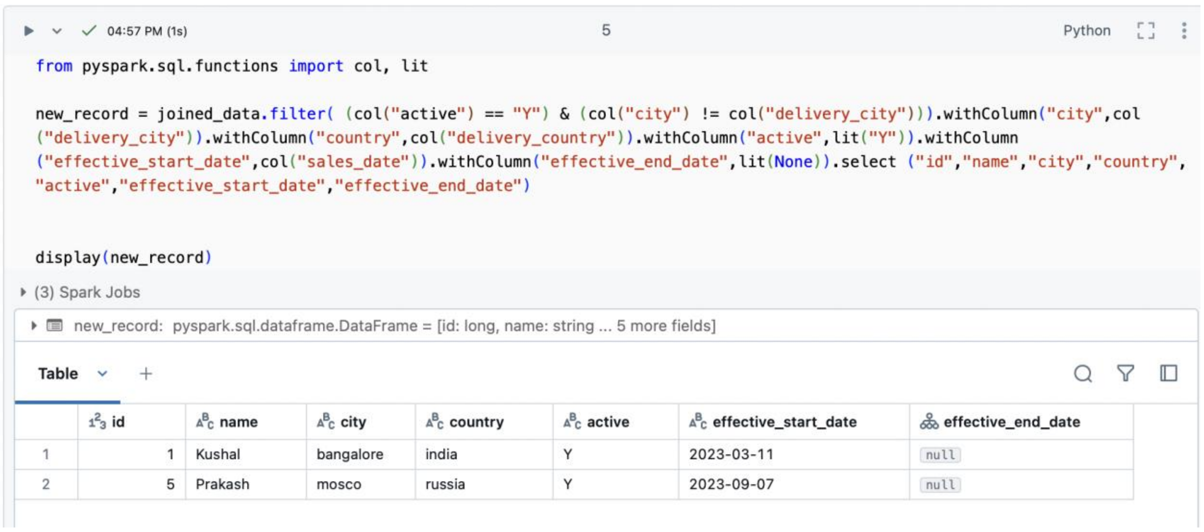
joined_data: pyspark.sql.dataframe.DataFrame = [id: long, name: string ... 12 more fields]

	id	name	city	country	active	effective_start_date	effective_end_da...	sales_id	cus
1	1	Kushal	assam	india	N	2022-09-15	2022-09-25	77	
2	1	Kushal	assam	india	N	2022-09-15	2022-09-25	1	
3	2	vikash	patna	india	Y	2023-08-12	null	null	
4	3	Sachin	delhi	india	Y	2023-09-10	null	12	
5	4	Rinku	jaipur	india	Y	2023-06-10	null	54	
6	5	Prakash	NY	USA	Y	2023-06-10	null	65	
7	1	Kushal	gurgaon	india	Y	2022-09-25	null	77	
8	1	Kushal	gurgaon	india	Y	2022-09-25	null	1	

8 rows | 1.29 seconds runtime

Refreshed 20 minutes ago

Identify customers purchased from New Address, but there Dimention data is not updated



04:57 PM (1s) 5 Python

```python
from pyspark.sql.functions import col, lit

new_record = joined_data.filter( (col("active") == "Y") & (col("city") != col("delivery_city"))).withColumn("city",col
("delivery_city")).withColumn("country",col("delivery_country")).withColumn("active",lit("Y")).withColumn
("effective_start_date",col("sales_date")).withColumn("effective_end_date",lit(None)).select ("id","name","city","country",
"active","effective_start_date","effective_end_date")


display(new_record)
```

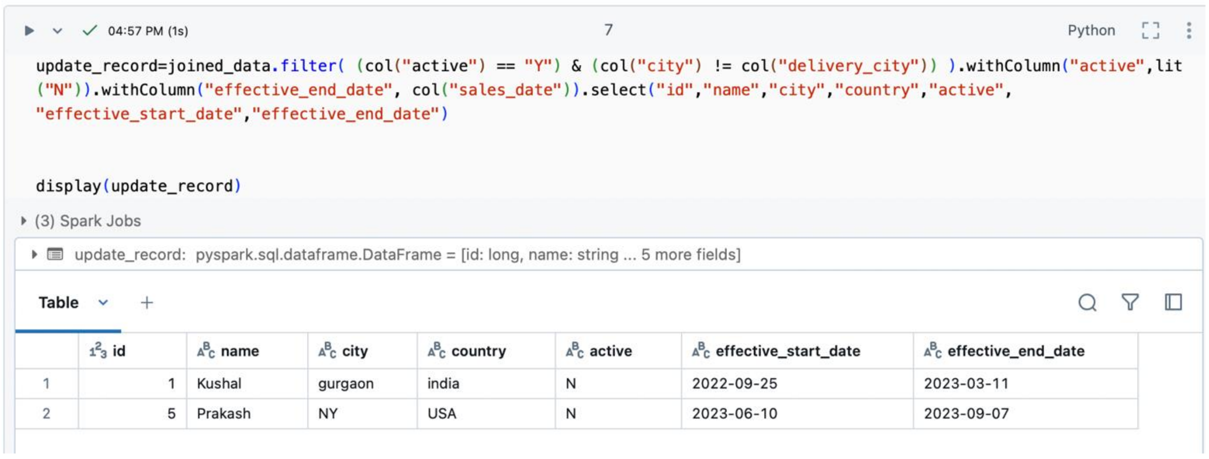
(3) Spark Jobs

new_record: pyspark.sql.dataframe.DataFrame = [id: long, name: string ... 5 more fields]

Table

	id	name	city	country	active	effective_start_date	effective_end_date
1	1	Kushal	bangalore	india	Y	2023-03-11	null
2	5	Prakash	mosco	russia	Y	2023-09-07	null

De-active the active FLAG and update the effective_end_date in the customer Dimension table



04:57 PM (1s) 7 Python

```python
update_record=joined_data.filter( (col("active") == "Y") & (col("city") != col("delivery_city")) ).withColumn("active",lit
("N")).withColumn("effective_end_date", col("sales_date")).select("id","name","city","country","active",
"effective_start_date","effective_end_date")


display(update_record)
```

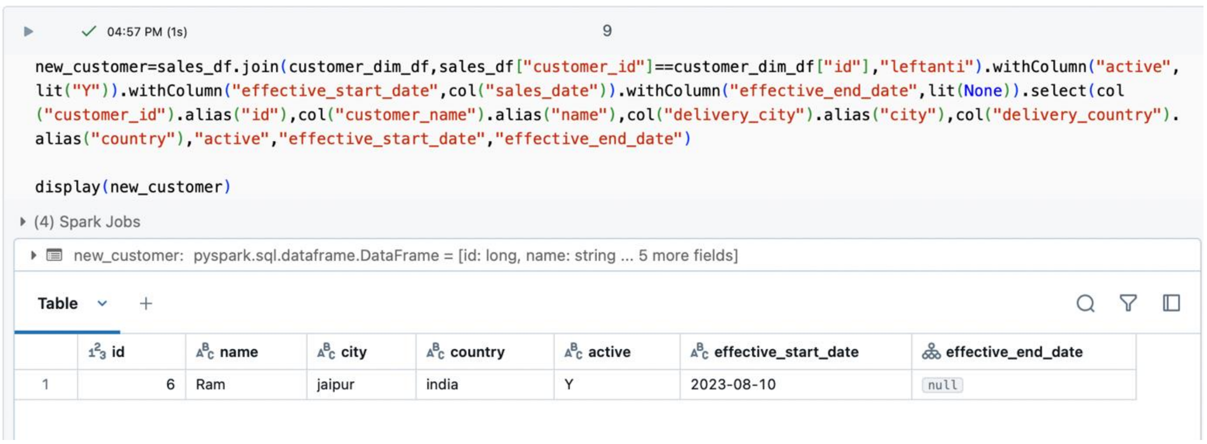
(3) Spark Jobs

update_record: pyspark.sql.dataframe.DataFrame = [id: long, name: string ... 5 more fields]

Table

	id	name	city	country	active	effective_start_date	effective_end_date
1	1	Kushal	gurgaon	india	N	2022-09-25	2023-03-11
2	5	Prakash	NY	USA	N	2023-06-10	2023-09-07

New Customer, whose data is not available in the Customer dimension



04:57 PM (1s)

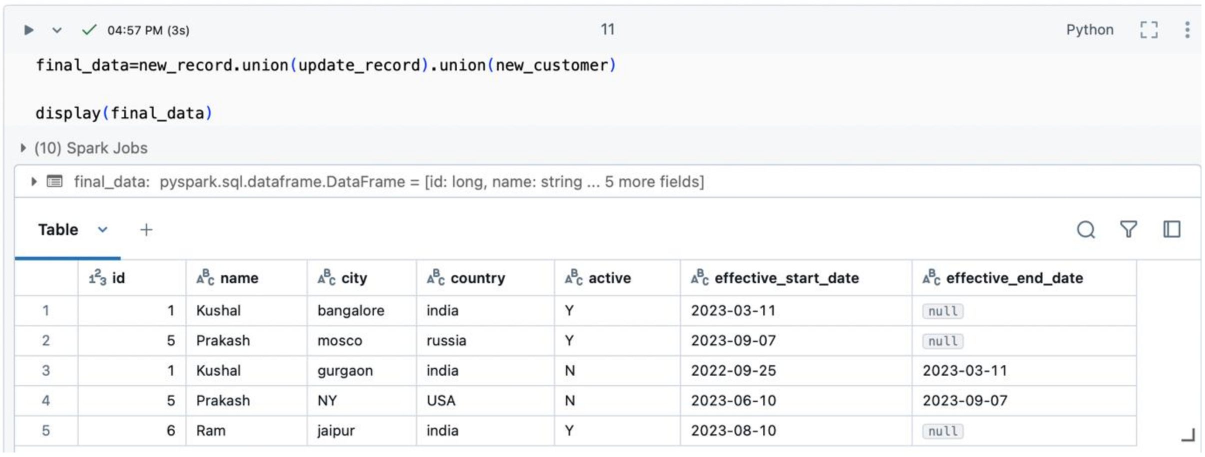
```
new_customer=sales_df.join(customer_dim_df,sales_df["customer_id"]==customer_dim_df["id"],"leftanti").withColumn("active",
lit("Y")).withColumn("effective_start_date",col("sales_date")).withColumn("effective_end_date",lit(None)).select(col
("customer_id").alias("id"),col("customer_name").alias("name"),col("delivery_city").alias("city"),col("delivery_country").
alias("country"),"active","effective_start_date","effective_end_date")

display(new_customer)
```

(4) Spark Jobs

new_customer: pyspark.sql.dataframe.DataFrame = [id: long, name: string ... 5 more fields]

	id	name	city	country	active	effective_start_date	effective_end_date
1	6	Ram	jaipur	india	Y	2023-08-10	null

Union all the data



04:57 PM (3s)

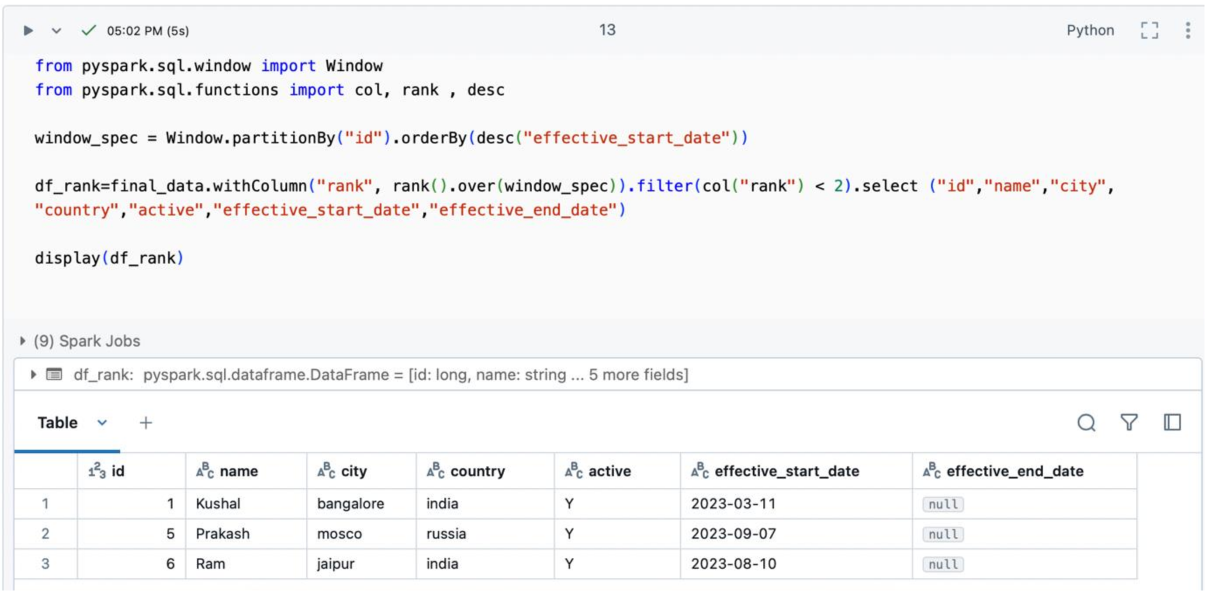
```
final_data=new_record.union(update_record).union(new_customer)

display(final_data)
```

(10) Spark Jobs

final_data: pyspark.sql.dataframe.DataFrame = [id: long, name: string ... 5 more fields]

	id	name	city	country	active	effective_start_date	effective_end_date
1	1	Kushal	bangalore	india	Y	2023-03-11	null
2	5	Prakash	mosco	russia	Y	2023-09-07	null
3	1	Kushal	gurgaon	india	N	2022-09-25	2023-03-11
4	5	Prakash	NY	USA	N	2023-06-10	2023-09-07
5	6	Ram	jaipur	india	Y	2023-08-10	null

Remove the duplicate records



```python
from pyspark.sql.window import Window
from pyspark.sql.functions import col, rank , desc

window_spec = Window.partitionBy("id").orderBy(desc("effective_start_date"))

df_rank=final_data.withColumn("rank", rank().over(window_spec)).filter(col("rank") < 2).select ("id","name","city",
"country","active","effective_start_date","effective_end_date")

display(df_rank)
```

df_rank: pyspark.sql.dataframe.DataFrame = [id: long, name: string ... 5 more fields]

	id	name	city	country	active	effective_start_date	effective_end_date
1	1	Kushal	bangalore	india	Y	2023-03-11	null
2	5	Prakash	mosco	russia	Y	2023-09-07	null
3	6	Ram	jaipur	india	Y	2023-08-10	null