Join function in pyspark

Join() function in pyspark

Join function is a SQL-like function in spark dialect, designed to merge or join two dataframe by function-base design interface.

The arguments in which the function set up initially are requied for retrieving different join/merge logic from respective dataframe.

These argument can be mainly distinguished by subtle differences into four types:

1. inner/left/right
2. outer - leftouter, rightouter, fullouter
3. semi/anti -leftsemi, leftanti
4. cross

Each type will be explained below

In [49]:
```
# prepare dataframes
from pyspark.sql.types import *
from pyspark.sql.types import StructField, StructType, IntegerType, StringType
df1 = spark.createDataFrame(
    [
        ('Eric',30,'Male'),
        ('Amy',28,'Female'),
        ('Lucy',31,'Female'),
        ('Lucas',32,'Male'),
        ('Ammie',25,'Female'),
    ],
    StructType([StructField("name", StringType(), True), StructField("Age", Integer
)
df2 = spark.createDataFrame(
    [
        ('Eric',32,'Male'),
        ('Lucy',31,'Female'),
        ('Lucas',32,'Male'),
        ('Judy',34,'Female'),
        ('Ammie',24,'Female'),
    ],
    StructType([StructField("name", StringType(), True), StructField("Age", Integer
)
```

```
StatementMeta(sparkcluster01, 9619, 2, Finished, Available)
```

In [50]:
```
display(df1)
```

```
StatementMeta(sparkcluster01, 9619, 3, Finished, Available)
SynapseWidget(Synapse.DataFrame, 1bd4b994-69c0-429d-9c6a-fb41f1857553)
```

In [51]:
```
display(df2)
```

```
StatementMeta(sparkcluster01, 9619, 4, Finished, Available)
SynapseWidget(Synapse.DataFrame, 393ada15-7828-4852-8594-5a34023345da)
```

inner / left/ right

[retrieve both dataframe datas]

- "inner" is simplest and default type of spark join function, inner join returns the result rows when matching column condition is met
- "left" join returns the result rows when matching column condition(s) is met and exist in left table only
- "right" join returns the result rows when matching column condition(s) is met and exist in right table only

In [52]:
```
display(
    df1.join(df2,'name','inner')
)
```

```
StatementMeta(sparkcluster01, 9619, 5, Finished, Available)
SynapseWidget(Synapse.DataFrame, 95bab885-7cf2-4dc0-a8bc-8cf31ffade22)
```

In [53]:
```
display(
    df1.join(df2,'name','left')
)
```

```
StatementMeta(sparkcluster01, 9619, 6, Finished, Available)
SynapseWidget(Synapse.DataFrame, c8a52f7e-311a-4ae6-b3fc-c9c1c485e5c2)
```

In [55]:
```
#outer 預設為fullouter
display(
    df1.join(df2,'name','right')

)
```

```
StatementMeta(sparkcluster01, 9619, 8, Finished, Available)
SynapseWidget(Synapse.DataFrame, 80559231-af68-4212-aa00-c782cf43f7e9)
```

leftouter / rightouter/ fullouter

[retrieve both dataframe datas]

- "leftouter" join is equivalent to 'left' join
- "rightouter" join is equivalent to 'right' join
- "fullouter" or "outer" join in pyspark combines the results of both left and right outer joins. The joined table will contain all records from both the tables

In [58]:
```
display(
    df1.join(df2,'name','outer')
)
```

```
StatementMeta(sparkcluster01, 9619, 11, Finished, Available)
SynapseWidget(Synapse.DataFrame, 9fd7ef69-e8e3-4299-a2b4-eb745f6ee135)
```

In [56]:
```
display(
    df1.join(df2,'name','leftouter')
)
```

```
StatementMeta(sparkcluster01, 9619, 9, Finished, Available)
SynapseWidget(Synapse.DataFrame, 7fc13d30-8732-43be-ab4d-2649fd0cbfa1)
```

In [59]:
```
display(
    df1.join(df2,'name','rightouter')
)
```

```
StatementMeta(sparkcluster01, 9619, 12, Finished, Available)
SynapseWidget(Synapse.DataFrame, 52826d78-7fb8-4ce6-858f-e244763f324b)
```

semi / anti/

[retrieve data from one table when column matched]

- "semi" or "leftsemi" join returns the result left table rows which are matched with wanted column and exist in left table only
- "anti" or "leftanti" join returns the result left table rows which are NOT matched with wanted column and exist in left table only
- "right" join returns the result rows when matching column condition(s) is met and exist in right table only

In [60]:
```
display(
    df1.join(df2,'name','semi')
)
```

```
StatementMeta(sparkcluster01, 9619, 13, Finished, Available)
SynapseWidget(Synapse.DataFrame, 54765e89-f480-416f-a2e3-cd1d999eb975)
```

In [62]:
```
display(
    df1.join(df2,'name','anti')
)
```

```
StatementMeta(sparkcluster01, 9619, 15, Finished, Available)
SynapseWidget(Synapse.DataFrame, e43c3a0b-ebaa-4181-8859-7e2f9b9f406a)
```

cross

[retrieve both dataframe datas]

- "cross" join returns Cartesian product result in which make engagement with two specify dataframe

In [69]:
```
display(
    df1.join(df2)
)
```

```
StatementMeta(sparkcluster01, 9619, 22, Finished, Available)
SynapseWidget(Synapse.DataFrame, 8866492b-e81c-4231-bdea-1b2e70c9608a)
```

The case engaging with multiple matching columns

In [72]:
```
display(
    df1.join(df2,['name','age'],'anti')
)
```

```
StatementMeta(sparkcluster01, 9619, 25, Finished, Available)
SynapseWidget(Synapse.DataFrame, c6c71ed1-ace5-425a-b962-794523d7e66f)
```