**Pushing Data to MySQL**

1. Get the Data from Redis. Continuously watch if data has existed. When data has existed then follow below steps
2. This will also be configured in Scheduler. Once WebSocket is running and data is fetched then this also should at same time. We need to uses same schedular details.
3. Data should get insert as quickly as possible.
4. First Data should insert in FIFO method only.
5. Once 1 record in inserted then only that row should be delete from Redis
6. Name of table should **data\_harvesting\_20241118** format (current day format YYYYMMDD)
7. LastUpdateTime should be parsed from UNIX timestamp to IST (Please check logic in indexfuture.py
8. Trading symbol from mapping those exchangeInstrumentID and then insert (Please ask logic to Deepak or in same (indexfuture.py)
9. Once all records Redis is inserted in Table then Check the records count of table and data inserted. Both should be same. Raise Error on Console if both are not matching.
10. Check **“insert\_trade\_data**” Method from indexfuture.py
11. Also add logic uk zone
12. async def insert\_trade\_data(data,combined\_token\_names):
13. try:
14. # Establish the database connection
15. #print(f'combined\_token\_names :{combined\_token\_names}')
16. connection = mysql.connector.connect(
17. \*\*DB\_CONNECTION\_PARAMS
18. )
19. table\_namenew = "data\_harvesting" + "\_" + datetime.now().strftime("%Y%m%d")
20. exchange\_id = data.get('ExchangeInstrumentID')
21. token\_name = None
22. time\_difference = 0
23. config = load\_config('config.json')
24. if config["timezone"].lower()=="uk":
25. time\_difference = 60
26. else:
27. time\_difference= 330
29. if isinstance(combined\_token\_names, dict):
30. token\_name = combined\_token\_names.get(exchange\_id, None)
31. elif isinstance(combined\_token\_names, list):
32. token\_name = next((item for item in combined\_token\_names if item.get('ExchangeInstrumentID') == exchange\_id), None)
34. if connection.is\_connected():
35. cursor = connection.cursor()
37. # SQL insert statement
38. insert\_query = f"""INSERT INTO {table\_namenew} (
39. MessageCode,
40. ExchangeSegment,
41. ExchangeInstrumentID,
42. BookType,
43. XMarketType,
44. LastTradedPrice,
45. LastTradedQuantity,
46. LastUpdateTime,
47. PercentChange,
48. Close,
49. Tradingsymbol,
50. OverallData
51. ) VALUES (%s, %s, %s, %s, %s, %s, %s, %s, %s, %s, %s, %s)"""
53. # Prepare the data as a tuple
54. record = (
55. data['MessageCode'],
56. data['ExchangeSegment'],
57. data['ExchangeInstrumentID'],
58. data['BookType'],
59. data['XMarketType'],
60. data['LastTradedPrice'],
61. data['LastTradedQunatity'],
62. datetime.fromtimestamp(data['LastUpdateTime'] + 315532800) - timedelta(minutes=time\_difference) if data['LastUpdateTime'] is not None else None,
63. data['PercentChange'],
64. data['Close'],
65. get\_token\_name(exchange\_id, combined\_token\_names),
66. json.dumps(data)  # Ensure item is a dictionary or JSON-serializable object
67. )
69. # Execute the insert statement
70. cursor.execute(insert\_query, record)
72. # Commit the transaction
73. connection.commit()
75. except mysql.connector.Error as error:
76. print(f"Failed to insert record: {error}")
77. finally:
78. if connection.is\_connected():
79. cursor.close()
80. connection.close()
81. #print("MySQL connection is closed")