ภาคผนวก ก

ชุดรหัสคำสั่งการวิเคราะห์ข้อมูลดิบที่นำเข้าจากโปรแกรม Syberiz ด้วยภาษา Python ในโปรแกรม Visual Studio Code

```
import pandas as pd
import time
import math
import time
from geopy.geocoders import Nominatim
from geopy.exc import GeocoderTimedOut

df = pd.read_csv('qqq.csv')
df = df.dropna(subset=['location'])
pattern = r'NBTC-BOX0[1-9]|NBTC-BOX1[0-9]|NBTC-BOX2[0-5]|NBTC-BOX3[0,1,4]|NBTC-BOX28'
filtered_data = df[df['location'].str.contains(pattern)]
df = filtered_data
df = df[df['location'].notna()]
start = time.time()
mapping_data = pd.read_excel('mapping_Book.xlsx')
mapping_df = pd.read_excel('map_Voice.xlsx')
additional_columns = [f'x{i}' for i in range(1, 12)]
df[additional_columns] = None
source_columns = mapping_df['source_column']
target_columns = mapping_df['target_column']
df = df[list(target_columns) + list(set(df.columns) - set(target_columns) - set(source_columns))]
column_mapping = mapping_df.set_index('source_column')['target_column'].to_dict()

df['START_TIME'] = pd.to_datetime(df['START_TIME'])
df.insert(df.columns.get_loc('START_TIME') + 1, 'TIME_ZONE', [''] * len(df))
time_ranges = [
    (pd.to_datetime('10:00').time(), pd.to_datetime('13:00').time(), 'DAY'),
    (pd.to_datetime('16:00').time(), pd.to_datetime('19:00').time(), 'NIGHT')
```

```
        ]
        for start_range, end_range, value in time_ranges:
            check = (df['START_TIME'].dt.time >= start_range) & (df['START_TIME'].dt.time < end_range)
            df.loc[check, 'TIME_ZONE'] = value
        df.loc[df['TIME_ZONE'] == '', 'TIME_ZONE'] = 'OTHERS'
        df.rename(columns={'band': 'BAND','TIME_ZONE':'FINISH_TIME'}, inplace=True)

        df.insert(df.columns.get_loc('location') + 1, 'location2', [''] * len(df))
        df.insert(df.columns.get_loc('location') + 4, 'BOX_TYPE', [''] * len(df))
        df.insert(df.columns.get_loc('NETTYPE') + 1, 'NETTYPE2', [''] * len(df))
        df.insert(df.columns.get_loc('MCCMNC') + 1, 'MCCMNC2', df['MCCMNC'])
        df.insert(df.columns.get_loc('geom') + 1, 'voc_status', '0')
        for index, row in mapping_data.iterrows():
            df.loc[(df['location'] == row['location']) , 'BOX_TYPE'] = row['BNUMBER3']
            df.loc[(df['BOX_TYPE'] == row['box_type']) & (df['operator1'] == row['operator1(voc)']) & (df['BNUMBER'] == row['bnumber']),'location2'] = row['voc_onoff']
            df.loc[(df['operator1'] == row['operator1(CP)']) & (df['BNUMBER'] == row['bnumber(CP)']), 'location2'] = row['voc(CP)']
            df.loc[(df['BNUMBER'] == row['fixed_line']),'location2'] = row['voc(CP)']
            df.loc[df['NETTYPE'] == row['NETTYPE'], 'NETTYPE2'] = row['NETTYPE2']
            df.loc[(df['operator1'] == row['operator1']) & (df['MCCMNC'] == row['MCCMNC']), 'MCCMNC2'] = row['MCCMNC2']
            df.loc[(df['BOX_TYPE'] == row['box_type']) & (df['operator1'] == row['operator1(voc)']) & (df['BNUMBER'] == row['bnumber']),'voc_status'] = row['voc_status']
        for index, row in mapping_data.iterrows():
            if not pd.isna(row['nr_band']) :
```

```
        df.loc[(df['NETTYPE'] == row['NETTYPE']) & (df['nr_band'] == row['nr_band']) , 'NETTYPE2'] = row['NETTYPE2']
df.loc[df['location2'] == '', 'location2'] = 'ANY'

geolocator = Nominatim(user_agent="reverse_geocoder")
cache = {}
def reverse_geocode_with_cache(lat, lng):
    if (lat, lng) in cache:
        return cache[(lat, lng)]
    try:
        location = geolocator.reverse((lat, lng), exactly_one=True)
        address = location.address if location else None
        district = None
        if address:
            address_components = address.split(',')
            for component in address_components:
                if 'จังหวัด' in component.lower() or 'กรุงเทพมหานคร' in component.lower():
                    district = component.strip()
                    break
        cache[(lat, lng)] = district
        return district
    except (AttributeError, GeocoderTimedOut):
        return None

df['CONNECTED_TIME'] = pd.to_datetime(df['CONNECTED_TIME'])
df['ALERTING_TIME'] = pd.to_datetime(df['ALERTING_TIME'])
df['DIALING_TIME'] = pd.to_datetime(df['DIALING_TIME'])
df['DISCONNECTED_TIME'] = pd.to_datetime(df['DISCONNECTED_TIME'])
df.insert(df.columns.get_loc('DISCONNECTED_TIME') + 1, 'DIALING_DURATION', None)
```

```
    df.insert(df.columns.get_loc('DISCONNECTED_TIME') + 2,
'ALERTING_DURATION', None)
    df.insert(df.columns.get_loc('DISCONNECTED_TIME') + 3,
'CONNECTED_DURATION', None)

    def one_time(time1,time2):
        if time1 == '' or time2 == '':
            return ''
        else:
            time = time2 - time1
            hours, remainder = divmod(time.total_seconds(), 3600)
            minutes, seconds = divmod(remainder, 60)
            return f"{int(hours):02}:{int(minutes):02}:{int(seconds):02}"

    for index, row in df.iterrows():
        disconnected_time = row['DISCONNECTED_TIME']
        connected_time = row['CONNECTED_TIME']
        alerting_time = row['ALERTING_TIME']
        dialing_time = row['DIALING_TIME']

        if pd.isna(dialing_time) or all(time == '' for time in [disconnected_time,
connected_time, alerting_time]):
            df.at[index, 'DIALING_DURATION'] = None
            df.at[index, 'ALERTING_DURATION'] = None
            df.at[index, 'CONNECTED_DURATION'] = None

        elif pd.notna(dialing_time):
            if pd.isna(disconnected_time):
                if pd.isna(connected_time) and pd.notna(alerting_time):
                    df.at[index, 'DIALING_DURATION'] = one_time(dialing_time,
alerting_time)
```

```
            elif pd.notna(connected_time) and pd.isna(alerting_time):
                df.at[index, 'DIALING_DURATION'] = one_time(dialing_time, connected_time)
            elif pd.notna(connected_time) and pd.notna(alerting_time):
                df.at[index, 'DIALING_DURATION'] = one_time(dialing_time, alerting_time)
                df.at[index, 'ALERTING_DURATION'] = one_time(alerting_time, connected_time)

        elif pd.isna(connected_time):
            if pd.notna(disconnected_time) and pd.isna(alerting_time):
                df.at[index, 'DIALING_DURATION'] = one_time(dialing_time, disconnected_time)
            elif pd.notna(disconnected_time) and pd.notna(alerting_time):
                df.at[index, 'DIALING_DURATION'] = one_time(dialing_time, alerting_time)
                df.at[index, 'ALERTING_DURATION'] = one_time(alerting_time, disconnected_time)

        elif pd.isna(alerting_time):
            if pd.notna(disconnected_time) and pd.notna(connected_time):
                df.at[index, 'DIALING_DURATION'] = one_time(dialing_time, connected_time)
                df.at[index, 'CONNECTED_DURATION'] = one_time(connected_time, disconnected_time)

        elif all(pd.notna(time) for time in [disconnected_time, connected_time, alerting_time, dialing_time]):
            df.at[index, 'DIALING_DURATION'] = one_time(dialing_time,alerting_time)
```

```
                    df.at[index, 'ALERTING_DURATION'] =
one_time(alerting_time,connected_time)
                    df.at[index, 'CONNECTED_DURATION'] =
one_time(connected_time,disconnected_time)

        df.insert(df.columns.get_loc('LNG') + 1, 'LATLNG2', [''] * len(df))
        for index, row in df.iterrows():
            mapping_row = mapping_data[mapping_data['location'] ==
row['location']]
            if not mapping_row.empty:
                mapping_row = mapping_row.iloc[0]
                if row['LAT'] != 0 and row['LNG'] != 0:
                    radius = 6371
                    dlat = math.radians(mapping_row['LAT2'] - row['LAT'])
                    dlon = math.radians(mapping_row['LNG2'] - row['LNG'])
                    a = math.sin(dlat / 2) ** 2 + math.cos(math.radians(row['LAT'])) *
math.cos(math.radians(mapping_row['LAT2'])) * math.sin(dlon / 2) ** 2
                    c = 2 * math.atan2(math.sqrt(a), math.sqrt(1 - a))
                    distance = radius * c * 1000
                if row['LAT'] == 0 and row['LNG'] == 0:
                    distance = 0
                if distance >= 100:
                    output = 'NO'
                if distance < 100:
                    output = 'YES'
                if distance == 0:
                    output = 'NO GPS'
            else:
                output = 'TEAM DRIVE TEST'
            df.at[index, 'LATLNG2'] = output
        for index, row in mapping_data.iterrows():
```

```
    df.loc[( (df['LATLNG2'] == row['LATLNG2']) & (df['location'] == row['location'])) , 'Address'] = row['Address']
for index, row in mapping_data.iterrows():
    df.loc[( (df['Address'].isna() ) & (df['location'] == row['location'])) , 'Address'] = row['Address']
print(df['Address'])

def check_duration(duration):
    duration = duration.split(':')
    hours, minutes, seconds = map(int, duration)
    duration_seconds = hours * 3600 + minutes * 60 + seconds
    return duration_seconds

def check_signal(row):
    if rssi <= -125:
        return ('Drop', 'RSSI <= -125') if status == 'PASS' else ('Block', 'RSSI <= -125')
    elif rssi > -125:
        return ('Bad', 'Unknown | PASS') if status == 'PASS' else ('Bad', 'Unknown | FAIL')
    else:
        return ('Bad', 'check rssi | PASS') if status == 'PASS' else ('Bad', 'check rssi | FAIL')

def check_nettype(row):
    if nettype2 == 'No service':
        return ('Drop', 'No service') if status == 'PASS' else ('Block', 'No service')
    elif nettype2 == 'UNKNOWN':
        if pd.isna(mccmnc):
            return ('Drop', 'No Nettype') if status == 'PASS' else ('Block', 'No Nettype')
        elif pd.notna(mccmnc):
            return ('Bad', 'Unknown Nettype | PASS') if status == 'PASS' else ('Bad', 'Unknown Nettype | FAIL')
    elif nettype2 in ['2G', '3G', '4G', '5G', '5G (2600 MHz)','5G (SA 2600 MHz)']:
        a, b = check_signal(row)
        return a, b
    else:
        return ('Bad', 'check nettype | PASS') if status == 'PASS' else ('Bad', 'check nettype | FAIL')

def check_cause_a_num(row):
    cause_a_num = row['CAUSE_A_NUM']
    if cause_a_num not in [-1, 1, 8, 16, 17, 31]:
        return ('Drop', 'Network') if status == 'PASS' else ('Block', 'Network')
    elif cause_a_num in [-1, 1, 8, 16, 17, 31]:
        a,b = check_nettype(row)
        return a, b
    else:
        return ('Bad', 'check network | PASS') if status == 'PASS' else ('Bad', 'check network | FAIL')

def check_callsetup(row):
    if call_setup == 0:
        return ('Bad', 'Call setup = 0 | PASS') if status == 'PASS' else ('Bad', 'Call setup = 0 | FAIL')
    elif call_setup != 0:
        a, b = check_cause_a_num(row)
        return a, b
    else:
        return ('Bad', 'check call setup | PASS') if status == 'PASS' else ('Bad', 'check call setup | FAIL')
```

```
def check_timeout(row):
    alerting_duration = row['ALERTING_DURATION']
    if check_duration(dialing_duration) >= 20:
        return ('Bad', 'Alerting Timeout | PASS') if status == 'PASS' else ('Bad', 'Alerting Timeout | FAIL')
    elif check_duration(dialing_duration) < 20:
        if pd.isna(alerting_duration) or check_duration(alerting_duration) < 20:
            a, b = check_callsetup(row)
            return a, b
        elif check_duration(alerting_duration) >= 20:
            return ('Bad', 'Connected Timeout | PASS') if status == 'PASS' else ('Bad', 'Connected Timeout | FAIL')
        elif check_duration(alerting_duration) < 0:
            return ('Bad', 'Negative Alert Time | PASS') if status == 'PASS' else ('Bad', 'Negative Alert Time | FAIL')
    else:
        return ('Bad', 'check duration | PASS') if status == 'PASS' else ('Bad', 'check duration | FAIL')

def check_success_call(connected_duration, time):
    if check_duration(connected_duration) > time+5:
        return 'Bad', 'Too much Time | PASS'
    elif time <= check_duration(connected_duration) <= (time + 5):
        return 'Complete', '-'
    elif check_duration(connected_duration) < time:
        return check_timeout(row)
    else:
        return 'Bad', 'Unknown | PASS'

for index, row in df.iterrows():
        status = row['STATUS']
        disconnected_time = row['DISCONNECTED_TIME']
        connected_duration = row['CONNECTED_DURATION']
        alerting_duration = row['ALERTING_DURATION']
        dialing_duration = row['DIALING_DURATION']
        cause_a_num = row['CAUSE_A_NUM']
        nettype2 = row['NETTYPE2']
        mccmnc = row['MCCMNC']
        rssi = row['RSSIDBM']
        call_setup = row['CALLSETUP']
        location = row['location']

        if status == 'PASS':
            if pd.isna(connected_duration):
                df.at[index, 'CAUSE_A_TXT'] = 'Bad'
                df.at[index, 'CAUSE_B_TXT'] = 'No Connect | PASS'
            elif pd.notna(connected_duration) and pd.notna(disconnected_time):
                if location in ['NBTC-BOX26', 'NBTC-BOX29','NBTC-BOX32' , 'NBTC-
BOX33']:
                    df.at[index, 'CAUSE_A_TXT'], df.at[index, 'CAUSE_B_TXT'] =
check_success_call(connected_duration, 90)
                else:
                    df.at[index, 'CAUSE_A_TXT'], df.at[index, 'CAUSE_B_TXT'] =
check_success_call(connected_duration, 50)
            else:
                df.at[index, 'CAUSE_A_TXT'], df.at[index, 'CAUSE_B_TXT'] = ('Bad',
'Unknown | PASS')

        elif status == 'FAIL':
            if pd.isna(dialing_duration):
                df.at[index, 'CAUSE_A_TXT'], df.at[index, 'CAUSE_B_TXT'] = ('No Time', 'No Dialing Duration | FAIL')
            elif pd.notna(dialing_duration):
                if check_duration(dialing_duration) < 0 :
                    df.at[index, 'CAUSE_A_TXT'], df.at[index, 'CAUSE_B_TXT'] = ('Bad', 'Negative Dial Time | FAIL')
                else:
                    df.at[index, 'CAUSE_A_TXT'], df.at[index, 'CAUSE_B_TXT'] = check_timeout(row)
            else:
                df.at[index, 'CAUSE_A_TXT'], df.at[index, 'CAUSE_B_TXT'] = ('Bad', 'Unknown | FAIL')

    df['DIALING_TIME'] = df['DIALING_TIME'].dt.strftime('%H:%M:%S')
    df['ALERTING_TIME'] = df['ALERTING_TIME'].dt.strftime('%H:%M:%S')
    df['CONNECTED_TIME'] = df['CONNECTED_TIME'].dt.strftime('%H:%M:%S')
    df['DISCONNECTED_TIME'] = df['DISCONNECTED_TIME'].dt.strftime('%H:%M:%S')
    df.rename(columns={'CAUSE_A_TXT': 'TP','BNUMBER':'FILE_PATH','BOX_TYPE':'FILE_PATH_STATUS','CAUSE_B_TXT':'TP2'}, inplace=True)

    df.insert(df.columns.get_loc('nr_band') + 1 , 'Signal_Strength', [''] * len(df))
    df.loc[(df['NETTYPE2'] == '2G') & ((df['RSSIDBM'].isna()) | (df['RSSIDBM'] == 0)), 'Signal_Strength'] = 'not clear'
    df.loc[(df['NETTYPE2'] == '3G') & ((df['RSCP'].isna()) | (df['RSCP'] == 0)),'Signal_Strength'] = 'not clear'
    df.loc[(df['NETTYPE2'] == '4G') & ((df['RSRP'].isna()) | (df['RSRP'] == 0)),'Signal_Strength'] = 'not clear'
    df.loc[(df['NETTYPE2'].str.contains('5G')) & ((df['nr_ssrsrp'].isna()) | (df['nr_ssrsrp'] == 0)),'Signal_Strength'] = 'not clear'
```

```
df.loc[df['NETTYPE2'] == 'No service', 'Signal_Strength'] = 'No service'
df.loc[df['NETTYPE2'] == 'UNKNOWN', 'Signal_Strength'] = 'UNKNOWN'
df.loc[df['Signal_Strength'] == '', 'Signal_Strength'] = 'not clear'

for index, row in mapping_data.iterrows():
    if row['NETTYPE2(SS)'] == '2G':
        df.loc[(df['NETTYPE2'] == row['NETTYPE2(SS)']) & (df['RSSIDBM'] >= row['SS_legend2'] ) & (df['RSSIDBM'] < row['SS_legend1']) , 'Signal_Strength'] = row['SS_level']
    elif row['NETTYPE2(SS)'] == '3G' :
        df.loc[(df['NETTYPE2'] == row['NETTYPE2(SS)']) & (df['RSCP'] >= row['SS_legend2'] ) & (df['RSCP'] < row['SS_legend1']) , 'Signal_Strength'] = row['SS_level']
    elif row['NETTYPE2(SS)'] == '4G' :
        df.loc[(df['NETTYPE2'] == row['NETTYPE2(SS)']) & (df['RSRP'] >= row['SS_legend2'] ) & (df['RSRP'] < row['SS_legend1']) , 'Signal_Strength'] = row['SS_level']
    else :
        df.loc[(df['NETTYPE2'] == row['NETTYPE2(SS)']) & (df['nr_ssrsrp'] >= row['SS_legend2'] ) & (df['nr_ssrsrp'] < row['SS_legend1']) , 'Signal_Strength'] = row['SS_level']

df.insert(df.columns.get_loc('nr_band') + 2 , 'Signal_Quality', [''] * len(df))
df.loc[(df['NETTYPE2'] == '3G') & ((df['ECIO'].isna()) | (df['ECIO'] == 0)),'Signal_Quality'] = 'not clear'
df.loc[(df['NETTYPE2'] == '4G') & ((df['SINR'].isna()) | (df['SINR'] == 0)),'Signal_Quality'] = 'not clear'
df.loc[(df['NETTYPE2'].str.contains('5G')) & ((df['nr_sssinr'].isna()) | (df['nr_sssinr'] == 0)),'Signal_Quality'] = 'not clear'
df.loc[df['NETTYPE2'] == 'No service', 'Signal_Quality'] = 'No service'
df.loc[df['NETTYPE2'] == 'UNKNOWN', 'Signal_Quality'] = 'UNKNOWN'
df.loc[df['Signal_Quality'] == '', 'Signal_Quality'] = 'not clear'
```

```
for index, row in mapping_data.iterrows():
    if row['NETTYPE2(SS)'] == '3G' :
        df.loc[(df['NETTYPE2'] == row['NETTYPE2(SS)']) & (df['ECIO'] >= row['SQ_legend2'] ) & (df['ECIO'] < row['SQ_legend1']) , 'Signal_Quality'] = row['SS_level']
    elif row['NETTYPE2(SS)'] == '4G' :
        df.loc[(df['NETTYPE2'] == row['NETTYPE2(SS)']) & (df['SINR'] >= row['SQ_legend2'] ) & (df['SINR'] < row['SQ_legend1']) , 'Signal_Quality'] = row['SS_level']
    else :
        df.loc[(df['NETTYPE2'] == row['NETTYPE2(SS)']) & (df['nr_sssinr'] >= row['SQ_legend2'] ) & (df['nr_sssinr'] < row['SQ_legend1']) , 'Signal_Quality'] = row['SS_level']

df.insert(df.columns.get_loc('x5') + 1 , 'x13', [''] * len(df))
df['x2'] = df['location']
df['x3'] = df['location2']
df['x4'] = df['operator1']
df['x5'] = df['TP']
df['x13'] = df['TP2']

columns_to_drop = ['location', 'location2' , 'operator1' , 'TP', 'TP2']
df.drop(columns=columns_to_drop, inplace=True, errors='ignore')
df.rename(columns={'x2': 'location', 'x3': 'location2', 'x4': 'operator1', 'x5' : 'TP' , 'x6' : 'ERROR2' , 'x7' : 'ERROR3','x13':'TP2'}, inplace=True)
columns_to_delete = ['x1', 'x8', 'x9', 'x10', 'x11']
df.drop(columns=columns_to_delete, inplace=True)

df.drop('LNG2', axis=1, inplace=True, errors='ignore')
df.drop('LAT2', axis=1, inplace=True, errors='ignore')
df.to_csv('QQQ.csv', index=False , encoding = 'utf-8-sig')
```

ภาคผนวก ข

ชุดรหัสคำสั่งการคำนวณและการแสดงผลข้อมูลที่ได้จาก Visual Studio Code ในโปรแกรม Power BI

ส่วนของการคำนวณการบริการประเภทข้อมูล

```
          AWN =
          VAR NoS_Ping = CALCULATE(COUNTROWS('Database'),'Database'[TYPE] =
"Ping",'Database'[operator1] = "TH GSM")
          VAR NoS_HTTP = CALCULATE(COUNTROWS('Database'),'Database'[TYPE] =
"HTTP Download",'Database'[operator1] = "TH GSM")
          VAR NoS_FTPUL = CALCULATE(COUNTROWS('Database'),'Database'[TYPE] =
"FTP Upload",'Database'[operator1] = "TH GSM")
          VAR NoS_FTPDL = CALCULATE(COUNTROWS('Database'),'Database'[TYPE] =
"FTP Download",'Database'[operator1] = "TH GSM")
          VAR PASS_ping = CALCULATE(COUNTROWS('Database'),'Database'[TYPE] =
"Ping",'Database'[operator1] = "TH GSM",'Database'[STATUS] = "PASS")
          VAR PASS_http = CALCULATE(COUNTROWS('Database'),'Database'[TYPE] =
"HTTP Download",'Database'[operator1] = "TH GSM",'Database'[STATUS] = "PASS")
          VAR PASS_ftpul = CALCULATE(COUNTROWS('Database'),'Database'[TYPE] =
"FTP Upload",'Database'[operator1] = "TH GSM",'Database'[STATUS] = "PASS")
          VAR PASS_ftpdl = CALCULATE(COUNTROWS('Database'),'Database'[TYPE] =
"FTP Download",'Database'[operator1] = "TH GSM",'Database'[STATUS] = "PASS")
          VAR FAIL_ping = CALCULATE(COUNTROWS('Database'),'Database'[TYPE] =
"Ping",'Database'[operator1] = "TH GSM",'Database'[STATUS] = "FAIL")
          VAR FAIL_http = CALCULATE(COUNTROWS('Database'),'Database'[TYPE] =
"HTTP Download",'Database'[operator1] = "TH GSM",'Database'[STATUS] = "FAIL")
          VAR FAIL_ftpul = CALCULATE(COUNTROWS('Database'),'Database'[TYPE] =
"FTP Upload",'Database'[operator1] = "TH GSM",'Database'[STATUS] = "FAIL")
          VAR FAIL_ftpdl = CALCULATE(COUNTROWS('Database'),'Database'[TYPE] =
"FTP Download",'Database'[operator1] = "TH GSM",'Database'[STATUS] = "FAIL")
          VAR TP_PASS_ping = CALCULATE(COUNTROWS('Database'),'Database'[TYPE]
= "Ping",'Database'[operator1] = "TH GSM",'Database'[STATUS] = "PASS",'Database'[TP2] =
"yes")
```

```
        VAR TP_PASS_http = CALCULATE(COUNTROWS('Database'),'Database'[TYPE]
= "HTTP Download",'Database'[operator1] = "TH GSM",'Database'[STATUS] =
"PASS",'Database'[TP2] = "yes")
        VAR TP_PASS_ftpul = CALCULATE(COUNTROWS('Database'),'Database'[TYPE]
= "FTP Upload",'Database'[operator1] = "TH GSM",'Database'[STATUS] =
"PASS",'Database'[TP2] = "yes")
        VAR TP_PASS_ftpdl = CALCULATE(COUNTROWS('Database'),'Database'[TYPE]
= "FTP Download",'Database'[operator1] = "TH GSM",'Database'[STATUS] =
"PASS",'Database'[TP2] = "yes")
        VAR Avg_TP_Ping =
CALCULATE(AVERAGE('Database'[TP_CAL]),'Database'[TYPE] =
"Ping",'Database'[operator1] = "TH GSM",'Database'[STATUS] = "PASS")
        VAR Avg_TP_HTTP =
CALCULATE(AVERAGE('Database'[USED_TIME]),'Database'[TYPE] = "HTTP
Download",'Database'[operator1] = "TH GSM",'Database'[STATUS] = "PASS")
        VAR Avg_TP_ftpul =
CALCULATE(AVERAGE('Database'[TP_CAL]),'Database'[TYPE] = "FTP
Upload",'Database'[operator1] = "TH GSM",'Database'[STATUS] = "PASS")
        VAR Avg_TP_ftpdl =
CALCULATE(AVERAGE('Database'[TP_CAL]),'Database'[TYPE] = "FTP
Download",'Database'[operator1] = "TH GSM",'Database'[STATUS] = "PASS")
        VAR Avg_TP_web =
CALCULATE(AVERAGE('Database'[TP_CAL]),'Database'[TYPE] = "HTTP
Download",'Database'[operator1] = "TH GSM",'Database'[STATUS] = "PASS")
        RETURN
        SWITCH(
          SELECTEDVALUE('QoS_Parameter'[Index]),
          6, FORMAT(NoS_Ping, "#,##0"),
          17, FORMAT(NoS_HTTP, "#,##0"),
          27, FORMAT(NoS_FTPUL, "#,##0"),
          36, FORMAT(NoS_FTPDL, "#,##0"),
        5, FORMAT(PASS_ping, "#,##0"),
        11, FORMAT(PASS_http, "#,##0"),
        20, FORMAT(PASS_ftpul, "#,##0"),
        29, FORMAT(PASS_ftpdl, "#,##0"),
        13, FORMAT(FAIL_http, "#,##0"),
        22, FORMAT(FAIL_ftpul, "#,##0"),
        31, FORMAT(FAIL_ftpdl, "#,##0"),
        2, FORMAT(TP_PASS_ping,"#,##0"),
        15, FORMAT(TP_PASS_http, "#,##0"),
        24, FORMAT(TP_PASS_ftpul,"#,##0"),
        33, FORMAT(TP_PASS_ftpdl,"#,##0"),
        12, IF(ISBLANK(NoS_HTTP),"N/A",FORMAT(100 *
DIVIDE(PASS_http,NoS_HTTP),"##0.00")),
        21, IF(ISBLANK(NoS_FTPUL),"N/A",FORMAT(100 *
DIVIDE(PASS_ftpul,NoS_FTPUL),"##0.00")),
        30, IF(ISBLANK(NoS_FTPDL),"N/A",FORMAT(100*
DIVIDE(PASS_ftpdl,NoS_FTPDL),"##0.00")),
        14, IF(ISBLANK(NoS_HTTP),"N/A",FORMAT(100*
DIVIDE(FAIL_http,NoS_HTTP),"##0.00")),
        23, IF(ISBLANK(NoS_FTPUL),"N/A",FORMAT(100*
DIVIDE(FAIL_ftpul,NoS_FTPUL),"##0.00")),
        32, IF(ISBLANK(NoS_FTPDL),"N/A",FORMAT(100*
DIVIDE(FAIL_ftpdl,NoS_FTPDL),"##0.00")),
        4, IF(ISBLANK(PASS_ping),"N/A",FORMAT(100*
DIVIDE(TP_PASS_ping,PASS_ping),"##0.00")),
        26, IF(ISBLANK(PASS_ftpul),"N/A",FORMAT(100*
DIVIDE(TP_PASS_ftpul,PASS_ftpul),"##0.00")),
        35, IF(ISBLANK(PASS_ftpdl),"N/A",FORMAT(100*
DIVIDE(TP_PASS_ftpdl,PASS_ftpdl),"##0.00")),
        3, FORMAT(Avg_TP_Ping,"##0"),
        16, FORMAT(Avg_TP_HTTP / 1000,"#,##0.000"),
          25, FORMAT(Avg_TP_ftpul / 1000,"#,##0.000"),
          34, FORMAT(Avg_TP_ftpdl / 1000 ,"#,##0.000"),
          7, FORMAT(FAIL_ping,"#,##0"),
          8, IF(ISBLANK(NoS_Ping),"N/A",FORMAT(100 *
DIVIDE(FAIL_ping,NoS_Ping),"##0.00")),
          9, IF(ISBLANK(NoS_Ping),"N/A",FORMAT(100 *
DIVIDE(PASS_ping,NoS_Ping),"##0.00")),
          18, FORMAT(Avg_TP_web / 1000 ,"#,##0.000"),
          BLANK()
        )

        DTN =
        VAR NoS_Ping = CALCULATE(COUNTROWS('Database'),'Database'[TYPE] =
"Ping",'Database'[operator1] = "DTAC")
        VAR NoS_HTTP = CALCULATE(COUNTROWS('Database'),'Database'[TYPE] =
"HTTP Download",'Database'[operator1] = "DTAC")
        VAR NoS_FTPUL = CALCULATE(COUNTROWS('Database'),'Database'[TYPE] =
"FTP Upload",'Database'[operator1] = "DTAC")
        VAR NoS_FTPDL = CALCULATE(COUNTROWS('Database'),'Database'[TYPE] =
"FTP Download",'Database'[operator1] = "DTAC")
        VAR PASS_ping = CALCULATE(COUNTROWS('Database'),'Database'[TYPE] =
"Ping",'Database'[operator1] = "DTAC",'Database'[STATUS] = "PASS")
        VAR PASS_http = CALCULATE(COUNTROWS('Database'),'Database'[TYPE] =
"HTTP Download",'Database'[operator1] = "DTAC",'Database'[STATUS] = "PASS")
        VAR PASS_ftpul = CALCULATE(COUNTROWS('Database'),'Database'[TYPE] =
"FTP Upload",'Database'[operator1] = "DTAC",'Database'[STATUS] = "PASS")
        VAR PASS_ftpdl = CALCULATE(COUNTROWS('Database'),'Database'[TYPE] =
"FTP Download",'Database'[operator1] = "DTAC",'Database'[STATUS] = "PASS")
        VAR FAIL_ping = CALCULATE(COUNTROWS('Database'),'Database'[TYPE] =
"Ping",'Database'[operator1] = "DTAC",'Database'[STATUS] = "FAIL")
```

```
    VAR FAIL_http = CALCULATE(COUNTROWS('Database'),'Database'[TYPE] = "HTTP Download",'Database'[operator1] = "DTAC",'Database'[STATUS] = "FAIL")
    VAR FAIL_ftpul = CALCULATE(COUNTROWS('Database'),'Database'[TYPE] = "FTP Upload",'Database'[operator1] = "DTAC",'Database'[STATUS] = "FAIL")
    VAR FAIL_ftpdl = CALCULATE(COUNTROWS('Database'),'Database'[TYPE] = "FTP Download",'Database'[operator1] = "DTAC",'Database'[STATUS] = "FAIL")
    VAR TP_PASS_ping = CALCULATE(COUNTROWS('Database'),'Database'[TYPE] = "Ping",'Database'[operator1] = "DTAC",'Database'[STATUS] = "PASS",'Database'[TP2] = "yes")
    VAR TP_PASS_http = CALCULATE(COUNTROWS('Database'),'Database'[TYPE] = "HTTP Download",'Database'[operator1] = "DTAC",'Database'[STATUS] = "PASS",'Database'[TP2] = "yes")
    VAR TP_PASS_ftpul = CALCULATE(COUNTROWS('Database'),'Database'[TYPE] = "FTP Upload",'Database'[operator1] = "DTAC",'Database'[STATUS] = "PASS",'Database'[TP2] = "yes")
    VAR TP_PASS_ftpdl = CALCULATE(COUNTROWS('Database'),'Database'[TYPE] = "FTP Download",'Database'[operator1] = "DTAC",'Database'[STATUS] = "PASS",'Database'[TP2] = "yes")
    VAR Avg_TP_Ping = CALCULATE(AVERAGE('Database'[TP_CAL]),'Database'[TYPE] = "Ping",'Database'[operator1] = "DTAC",'Database'[STATUS] = "PASS")
    VAR Avg_TP_HTTP = CALCULATE(AVERAGE('Database'[USED_TIME]),'Database'[TYPE] = "HTTP Download",'Database'[operator1] = "DTAC",'Database'[STATUS] = "PASS")
    VAR Avg_TP_ftpul = CALCULATE(AVERAGE('Database'[TP_CAL]),'Database'[TYPE] = "FTP Upload",'Database'[operator1] = "DTAC",'Database'[STATUS] = "PASS")
    VAR Avg_TP_ftpdl = CALCULATE(AVERAGE('Database'[TP_CAL]),'Database'[TYPE] = "FTP Download",'Database'[operator1] = "DTAC",'Database'[STATUS] = "PASS")
```

```
        VAR Avg_TP_web =
CALCULATE(AVERAGE('Database'[TP_CAL]),'Database'[TYPE] = "HTTP
Download",'Database'[operator1] = "DTAC",'Database'[STATUS] = "PASS")
        RETURN
        SWITCH(
          SELECTEDVALUE('QoS_Parameter'[Index]),
          6, FORMAT(NoS_Ping, "#,##0"),
          17, FORMAT(NoS_HTTP, "#,##0"),
          27, FORMAT(NoS_FTPUL, "#,##0"),
          36, FORMAT(NoS_FTPDL, "#,##0"),
          5, FORMAT(PASS_ping, "#,##0"),
          11, FORMAT(PASS_http, "#,##0"),
          20, FORMAT(PASS_ftpul, "#,##0"),
          29, FORMAT(PASS_ftpdl, "#,##0"),
          13, FORMAT(FAIL_http, "#,##0"),
          22, FORMAT(FAIL_ftpul, "#,##0"),
          31, FORMAT(FAIL_ftpdl, "#,##0"),
          2, FORMAT(TP_PASS_ping,"#,##0"),
          15, FORMAT(TP_PASS_http, "#,##0"),
          24, FORMAT(TP_PASS_ftpul,"#,##0"),
          33, FORMAT(TP_PASS_ftpdl,"#,##0"),
          12, IF(ISBLANK(NoS_HTTP),"N/A",FORMAT(100 *
DIVIDE(PASS_http,NoS_HTTP),"##0.00")),
          21, IF(ISBLANK(NoS_FTPUL),"N/A",FORMAT(100 *
DIVIDE(PASS_ftpul,NoS_FTPUL),"##0.00")),
          30, IF(ISBLANK(NoS_FTPDL),"N/A",FORMAT(100*
DIVIDE(PASS_ftpdl,NoS_FTPDL),"##0.00")),
          14, IF(ISBLANK(NoS_HTTP),"N/A",FORMAT(100*
DIVIDE(FAIL_http,NoS_HTTP),"##0.00")),
          23, IF(ISBLANK(NoS_FTPUL),"N/A",FORMAT(100*
DIVIDE(FAIL_ftpul,NoS_FTPUL),"##0.00")),
          32, IF(ISBLANK(NoS_FTPDL),"N/A",FORMAT(100*
DIVIDE(FAIL_ftpdl,NoS_FTPDL),"##0.00")),
          4, IF(ISBLANK(PASS_ping),"N/A",FORMAT(100*
DIVIDE(TP_PASS_ping,PASS_ping),"##0.00")),
          26, IF(ISBLANK(PASS_ftpul),"N/A",FORMAT(100*
DIVIDE(TP_PASS_ftpul,PASS_ftpul),"##0.00")),
          35, IF(ISBLANK(PASS_ftpdl),"N/A",FORMAT(100*
DIVIDE(TP_PASS_ftpdl,PASS_ftpdl),"##0.00")),
          3, FORMAT(Avg_TP_Ping,"##0"),
          16, FORMAT(Avg_TP_HTTP / 1000,"#,##0.000"),
          25, FORMAT(Avg_TP_ftpul / 1000,"#,##0.000"),
          34, FORMAT(Avg_TP_ftpdl / 1000 ,"#,##0.000"),
          7, FORMAT(FAIL_ping,"#,##0"),
          8, IF(ISBLANK(NoS_Ping),"N/A",FORMAT(100 *
DIVIDE(FAIL_ping,NoS_Ping),"##0.00")),
          9, IF(ISBLANK(NoS_Ping),"N/A",FORMAT(100 *
DIVIDE(PASS_ping,NoS_Ping),"##0.00")),
          18, FORMAT(Avg_TP_web / 1000 ,"#,##0.000"),
          BLANK()
        )


        NT-CAT =
        VAR NoS_Ping = CALCULATE(COUNTROWS('Database'),'Database'[TYPE] =
"Ping",'Database'[operator1] = "my")
        VAR NoS_HTTP = CALCULATE(COUNTROWS('Database'),'Database'[TYPE] =
"HTTP Download",'Database'[operator1] = "my")
        VAR NoS_FTPUL = CALCULATE(COUNTROWS('Database'),'Database'[TYPE] =
"FTP Upload",'Database'[operator1] = "my")
        VAR NoS_FTPDL = CALCULATE(COUNTROWS('Database'),'Database'[TYPE] =
"FTP Download",'Database'[operator1] = "my")
```

```
VAR PASS_ping = CALCULATE(COUNTROWS('Database'),'Database'[TYPE] = "Ping",'Database'[operator1] = "my",'Database'[STATUS] = "PASS")
VAR PASS_http = CALCULATE(COUNTROWS('Database'),'Database'[TYPE] = "HTTP Download",'Database'[operator1] = "my",'Database'[STATUS] = "PASS")
VAR PASS_ftpul = CALCULATE(COUNTROWS('Database'),'Database'[TYPE] = "FTP Upload",'Database'[operator1] = "my",'Database'[STATUS] = "PASS")
VAR PASS_ftpdl = CALCULATE(COUNTROWS('Database'),'Database'[TYPE] = "FTP Download",'Database'[operator1] = "my",'Database'[STATUS] = "PASS")
VAR FAIL_ping = CALCULATE(COUNTROWS('Database'),'Database'[TYPE] = "Ping",'Database'[operator1] = "my",'Database'[STATUS] = "FAIL")
VAR FAIL_http = CALCULATE(COUNTROWS('Database'),'Database'[TYPE] = "HTTP Download",'Database'[operator1] = "my",'Database'[STATUS] = "FAIL")
VAR FAIL_ftpul = CALCULATE(COUNTROWS('Database'),'Database'[TYPE] = "FTP Upload",'Database'[operator1] = "my",'Database'[STATUS] = "FAIL")
VAR FAIL_ftpdl = CALCULATE(COUNTROWS('Database'),'Database'[TYPE] = "FTP Download",'Database'[operator1] = "my",'Database'[STATUS] = "FAIL")
VAR TP_PASS_ping = CALCULATE(COUNTROWS('Database'),'Database'[TYPE] = "Ping",'Database'[operator1] = "my",'Database'[STATUS] = "PASS",'Database'[TP2] = "yes")
VAR TP_PASS_http = CALCULATE(COUNTROWS('Database'),'Database'[TYPE] = "HTTP Download",'Database'[operator1] = "my",'Database'[STATUS] = "PASS",'Database'[TP2] = "yes")
VAR TP_PASS_ftpul = CALCULATE(COUNTROWS('Database'),'Database'[TYPE] = "FTP Upload",'Database'[operator1] = "my",'Database'[STATUS] = "PASS",'Database'[TP2] = "yes")
VAR TP_PASS_ftpdl = CALCULATE(COUNTROWS('Database'),'Database'[TYPE] = "FTP Download",'Database'[operator1] = "my",'Database'[STATUS] = "PASS",'Database'[TP2] = "yes")
VAR Avg_TP_Ping = CALCULATE(AVERAGE('Database'[TP_CAL]),'Database'[TYPE] = "Ping",'Database'[operator1] = "my",'Database'[STATUS] = "PASS")
```

```
    VAR Avg_TP_HTTP =
CALCULATE(AVERAGE('Database'[USED_TIME]),'Database'[TYPE] = "HTTP
Download",'Database'[operator1] = "my",'Database'[STATUS] = "PASS")
    VAR Avg_TP_ftpul =
CALCULATE(AVERAGE('Database'[TP_CAL]),'Database'[TYPE] = "FTP
Upload",'Database'[operator1] = "my",'Database'[STATUS] = "PASS")
    VAR Avg_TP_ftpdl =
CALCULATE(AVERAGE('Database'[TP_CAL]),'Database'[TYPE] = "FTP
Download",'Database'[operator1] = "my",'Database'[STATUS] = "PASS")
    VAR Avg_TP_web =
CALCULATE(AVERAGE('Database'[TP_CAL]),'Database'[TYPE] = "HTTP
Download",'Database'[operator1] = "my",'Database'[STATUS] = "PASS")
    RETURN
    SWITCH(
      SELECTEDVALUE('QoS_Parameter'[Index]),
      6, FORMAT(NoS_Ping, "#,##0"),
      17, FORMAT(NoS_HTTP, "#,##0"),
      27, FORMAT(NoS_FTPUL, "#,##0"),
      36, FORMAT(NoS_FTPDL, "#,##0"),
      5, FORMAT(PASS_ping, "#,##0"),
      11, FORMAT(PASS_http, "#,##0"),
      20, FORMAT(PASS_ftpul, "#,##0"),
      29, FORMAT(PASS_ftpdl, "#,##0"),
      13, FORMAT(FAIL_http, "#,##0"),
      22, FORMAT(FAIL_ftpul, "#,##0"),
      31, FORMAT(FAIL_ftpdl, "#,##0"),
      2, FORMAT(TP_PASS_ping,"#,##0"),
      15, FORMAT(TP_PASS_http, "#,##0"),
      24, FORMAT(TP_PASS_ftpul,"#,##0"),
      33, FORMAT(TP_PASS_ftpdl,"#,##0"),
            12, IF(ISBLANK(NoS_HTTP),"N/A",FORMAT(100 *
DIVIDE(PASS_http,NoS_HTTP),"##0.00")),
            21, IF(ISBLANK(NoS_FTPUL),"N/A",FORMAT(100 *
DIVIDE(PASS_ftpul,NoS_FTPUL),"##0.00")),
            30, IF(ISBLANK(NoS_FTPDL),"N/A",FORMAT(100*
DIVIDE(PASS_ftpdl,NoS_FTPDL),"##0.00")),
            14, IF(ISBLANK(NoS_HTTP),"N/A",FORMAT(100*
DIVIDE(FAIL_http,NoS_HTTP),"##0.00")),
            23, IF(ISBLANK(NoS_FTPUL),"N/A",FORMAT(100*
DIVIDE(FAIL_ftpul,NoS_FTPUL),"##0.00")),
            32, IF(ISBLANK(NoS_FTPDL),"N/A",FORMAT(100*
DIVIDE(FAIL_ftpdl,NoS_FTPDL),"##0.00")),
            4, IF(ISBLANK(PASS_ping),"N/A",FORMAT(100*
DIVIDE(TP_PASS_ping,PASS_ping),"##0.00")),
            26, IF(ISBLANK(PASS_ftpul),"N/A",FORMAT(100*
DIVIDE(TP_PASS_ftpul,PASS_ftpul),"##0.00")),
            35, IF(ISBLANK(PASS_ftpdl),"N/A",FORMAT(100*
DIVIDE(TP_PASS_ftpdl,PASS_ftpdl),"##0.00")),
            3, FORMAT(Avg_TP_Ping,"##0"),
            16, FORMAT(Avg_TP_HTTP / 1000,"#,##0.000"),
            25, FORMAT(Avg_TP_ftpul / 1000,"#,##0.000"),
            34, FORMAT(Avg_TP_ftpdl / 1000 ,"#,##0.000"),
            7, FORMAT(FAIL_ping,"#,##0"),
            8, IF(ISBLANK(NoS_Ping),"N/A",FORMAT(100 *
DIVIDE(FAIL_ping,NoS_Ping),"##0.00")),
            9, IF(ISBLANK(NoS_Ping),"N/A",FORMAT(100 *
DIVIDE(PASS_ping,NoS_Ping),"##0.00")),
            18, FORMAT(Avg_TP_web / 1000 ,"#,##0.000"),
            BLANK()
        )
```

```
NT-TOT =
VAR NoS_Ping = CALCULATE(COUNTROWS('Database'),'Database'[TYPE] = "Ping",'Database'[operator1] = "TOT")
VAR NoS_HTTP = CALCULATE(COUNTROWS('Database'),'Database'[TYPE] = "HTTP Download",'Database'[operator1] = "TOT")
VAR NoS_FTPUL = CALCULATE(COUNTROWS('Database'),'Database'[TYPE] = "FTP Upload",'Database'[operator1] = "TOT")
VAR NoS_FTPDL = CALCULATE(COUNTROWS('Database'),'Database'[TYPE] = "FTP Download",'Database'[operator1] = "TOT")
VAR PASS_ping = CALCULATE(COUNTROWS('Database'),'Database'[TYPE] = "Ping",'Database'[operator1] = "TOT",'Database'[STATUS] = "PASS")
VAR PASS_http = CALCULATE(COUNTROWS('Database'),'Database'[TYPE] = "HTTP Download",'Database'[operator1] = "TOT",'Database'[STATUS] = "PASS")
VAR PASS_ftpul = CALCULATE(COUNTROWS('Database'),'Database'[TYPE] = "FTP Upload",'Database'[operator1] = "TOT",'Database'[STATUS] = "PASS")
VAR PASS_ftpdl = CALCULATE(COUNTROWS('Database'),'Database'[TYPE] = "FTP Download",'Database'[operator1] = "TOT",'Database'[STATUS] = "PASS")
VAR FAIL_ping = CALCULATE(COUNTROWS('Database'),'Database'[TYPE] = "Ping",'Database'[operator1] = "TOT",'Database'[STATUS] = "FAIL")
VAR FAIL_http = CALCULATE(COUNTROWS('Database'),'Database'[TYPE] = "HTTP Download",'Database'[operator1] = "TOT",'Database'[STATUS] = "FAIL")
VAR FAIL_ftpul = CALCULATE(COUNTROWS('Database'),'Database'[TYPE] = "FTP Upload",'Database'[operator1] = "TOT",'Database'[STATUS] = "FAIL")
VAR FAIL_ftpdl = CALCULATE(COUNTROWS('Database'),'Database'[TYPE] = "FTP Download",'Database'[operator1] = "TOT",'Database'[STATUS] = "FAIL")
VAR TP_PASS_ping = CALCULATE(COUNTROWS('Database'),'Database'[TYPE] = "Ping",'Database'[operator1] = "TOT",'Database'[STATUS] = "PASS",'Database'[TP2] = "yes")
VAR TP_PASS_http = CALCULATE(COUNTROWS('Database'),'Database'[TYPE] = "HTTP Download",'Database'[operator1] = "TOT",'Database'[STATUS] = "PASS",'Database'[TP2] = "yes")
```

```
        VAR TP_PASS_ftpul = CALCULATE(COUNTROWS('Database'),'Database'[TYPE]
= "FTP Upload",'Database'[operator1] = "TOT",'Database'[STATUS] =
"PASS",'Database'[TP2] = "yes")
        VAR TP_PASS_ftpdl = CALCULATE(COUNTROWS('Database'),'Database'[TYPE]
= "FTP Download",'Database'[operator1] = "TOT",'Database'[STATUS] =
"PASS",'Database'[TP2] = "yes")
        VAR Avg_TP_Ping =
CALCULATE(AVERAGE('Database'[TP_CAL]),'Database'[TYPE] =
"Ping",'Database'[operator1] = "TOT",'Database'[STATUS] = "PASS")
        VAR Avg_TP_HTTP =
CALCULATE(AVERAGE('Database'[USED_TIME]),'Database'[TYPE] = "HTTP
Download",'Database'[operator1] = "TOT",'Database'[STATUS] = "PASS")
        VAR Avg_TP_ftpul =
CALCULATE(AVERAGE('Database'[TP_CAL]),'Database'[TYPE] = "FTP
Upload",'Database'[operator1] = "TOT",'Database'[STATUS] = "PASS")
        VAR Avg_TP_ftpdl =
CALCULATE(AVERAGE('Database'[TP_CAL]),'Database'[TYPE] = "FTP
Download",'Database'[operator1] = "TOT",'Database'[STATUS] = "PASS")
        VAR Avg_TP_web =
CALCULATE(AVERAGE('Database'[TP_CAL]),'Database'[TYPE] = "HTTP
Download",'Database'[operator1] = "TOT",'Database'[STATUS] = "PASS")
        RETURN
        SWITCH(
            SELECTEDVALUE('QoS_Parameter'[Index]),
            6, FORMAT(NoS_Ping, "#,##0"),
            17, FORMAT(NoS_HTTP, "#,##0"),
            27, FORMAT(NoS_FTPUL, "#,##0"),
            36, FORMAT(NoS_FTPDL, "#,##0"),
            5, FORMAT(PASS_ping, "#,##0"),
            11, FORMAT(PASS_http, "#,##0"),
            20, FORMAT(PASS_ftpul, "#,##0"),
            29, FORMAT(PASS_ftpdl, "#,##0"),
            13, FORMAT(FAIL_http, "#,##0"),
            22, FORMAT(FAIL_ftpul, "#,##0"),
            31, FORMAT(FAIL_ftpdl, "#,##0"),
            2, FORMAT(TP_PASS_ping,"#,##0"),
            15, FORMAT(TP_PASS_http, "#,##0"),
            24, FORMAT(TP_PASS_ftpul,"#,##0"),
            33, FORMAT(TP_PASS_ftpdl,"#,##0"),
            12, IF(ISBLANK(NoS_HTTP),"N/A",FORMAT(100 *
DIVIDE(PASS_http,NoS_HTTP),"##0.00")),
            21, IF(ISBLANK(NoS_FTPUL),"N/A",FORMAT(100 *
DIVIDE(PASS_ftpul,NoS_FTPUL),"##0.00")),
            30, IF(ISBLANK(NoS_FTPDL),"N/A",FORMAT(100*
DIVIDE(PASS_ftpdl,NoS_FTPDL),"##0.00")),
            14, IF(ISBLANK(NoS_HTTP),"N/A",FORMAT(100*
DIVIDE(FAIL_http,NoS_HTTP),"##0.00")),
            23, IF(ISBLANK(NoS_FTPUL),"N/A",FORMAT(100*
DIVIDE(FAIL_ftpul,NoS_FTPUL),"##0.00")),
            32, IF(ISBLANK(NoS_FTPDL),"N/A",FORMAT(100*
DIVIDE(FAIL_ftpdl,NoS_FTPDL),"##0.00")),
            4, IF(ISBLANK(PASS_ping),"N/A",FORMAT(100*
DIVIDE(TP_PASS_ping,PASS_ping),"##0.00")),
            26, IF(ISBLANK(PASS_ftpul),"N/A",FORMAT(100*
DIVIDE(TP_PASS_ftpul,PASS_ftpul),"##0.00")),
            35, IF(ISBLANK(PASS_ftpdl),"N/A",FORMAT(100*
DIVIDE(TP_PASS_ftpdl,PASS_ftpdl),"##0.00")),
            3, FORMAT(Avg_TP_Ping,"##0"),
            16, FORMAT(Avg_TP_HTTP / 1000,"#,##0.000"),
            25, FORMAT(Avg_TP_ftpul / 1000,"#,##0.000"),
            34, FORMAT(Avg_TP_ftpdl / 1000 ,"#,##0.000"),
            7, FORMAT(FAIL_ping,"#,##0"),
            8, IF(ISBLANK(NoS_Ping),"N/A",FORMAT(100 *
DIVIDE(FAIL_ping,NoS_Ping),"##0.00")),
            9, IF(ISBLANK(NoS_Ping),"N/A",FORMAT(100 *
DIVIDE(PASS_ping,NoS_Ping),"##0.00")),
            18, FORMAT(Avg_TP_web / 1000 ,"#,##0.000"),
            BLANK()
        )

        TRUE-H =
        VAR NoS_Ping = CALCULATE(COUNTROWS('Database'),'Database'[TYPE] =
"Ping",'Database'[operator1] = "TRUE-H")
        VAR NoS_HTTP = CALCULATE(COUNTROWS('Database'),'Database'[TYPE] =
"HTTP Download",'Database'[operator1] = "TRUE-H")
        VAR NoS_FTPUL = CALCULATE(COUNTROWS('Database'),'Database'[TYPE] =
"FTP Upload",'Database'[operator1] = "TRUE-H")
        VAR NoS_FTPDL = CALCULATE(COUNTROWS('Database'),'Database'[TYPE] =
"FTP Download",'Database'[operator1] = "TRUE-H")
        VAR PASS_ping = CALCULATE(COUNTROWS('Database'),'Database'[TYPE] =
"Ping",'Database'[operator1] = "TRUE-H",'Database'[STATUS] = "PASS")
        VAR PASS_http = CALCULATE(COUNTROWS('Database'),'Database'[TYPE] =
"HTTP Download",'Database'[operator1] = "TRUE-H",'Database'[STATUS] = "PASS")
        VAR PASS_ftpul = CALCULATE(COUNTROWS('Database'),'Database'[TYPE] =
"FTP Upload",'Database'[operator1] = "TRUE-H",'Database'[STATUS] = "PASS")
        VAR PASS_ftpdl = CALCULATE(COUNTROWS('Database'),'Database'[TYPE] =
"FTP Download",'Database'[operator1] = "TRUE-H",'Database'[STATUS] = "PASS")
        VAR FAIL_ping = CALCULATE(COUNTROWS('Database'),'Database'[TYPE] =
"Ping",'Database'[operator1] = "TRUE-H",'Database'[STATUS] = "FAIL")
        VAR FAIL_http = CALCULATE(COUNTROWS('Database'),'Database'[TYPE] =
"HTTP Download",'Database'[operator1] = "TRUE-H",'Database'[STATUS] = "FAIL")
        VAR FAIL_ftpul = CALCULATE(COUNTROWS('Database'),'Database'[TYPE] =
"FTP Upload",'Database'[operator1] = "TRUE-H",'Database'[STATUS] = "FAIL")
```

```
	VAR FAIL_ftpdl = CALCULATE(COUNTROWS('Database'),'Database'[TYPE] = "FTP Download",'Database'[operator1] = "TRUE-H",'Database'[STATUS] = "FAIL")
	VAR TP_PASS_ping = CALCULATE(COUNTROWS('Database'),'Database'[TYPE] = "Ping",'Database'[operator1] = "TRUE-H",'Database'[STATUS] = "PASS",'Database'[TP2] = "yes")
	VAR TP_PASS_http = CALCULATE(COUNTROWS('Database'),'Database'[TYPE] = "HTTP Download",'Database'[operator1] = "TRUE-H",'Database'[STATUS] = "PASS",'Database'[TP2] = "yes")
	VAR TP_PASS_ftpul = CALCULATE(COUNTROWS('Database'),'Database'[TYPE] = "FTP Upload",'Database'[operator1] = "TRUE-H",'Database'[STATUS] = "PASS",'Database'[TP2] = "yes")
	VAR TP_PASS_ftpdl = CALCULATE(COUNTROWS('Database'),'Database'[TYPE] = "FTP Download",'Database'[operator1] = "TRUE-H",'Database'[STATUS] = "PASS",'Database'[TP2] = "yes")
	VAR Avg_TP_Ping = CALCULATE(AVERAGE('Database'[TP_CAL]),'Database'[TYPE] = "Ping",'Database'[operator1] = "TRUE-H",'Database'[STATUS] = "PASS")
	VAR Avg_TP_HTTP = CALCULATE(AVERAGE('Database'[USED_TIME]),'Database'[TYPE] = "HTTP Download",'Database'[operator1] = "TRUE-H",'Database'[STATUS] = "PASS")
	VAR Avg_TP_ftpul = CALCULATE(AVERAGE('Database'[TP_CAL]),'Database'[TYPE] = "FTP Upload",'Database'[operator1] = "TRUE-H",'Database'[STATUS] = "PASS")
	VAR Avg_TP_ftpdl = CALCULATE(AVERAGE('Database'[TP_CAL]),'Database'[TYPE] = "FTP Download",'Database'[operator1] = "TRUE-H",'Database'[STATUS] = "PASS")
	VAR Avg_TP_web = CALCULATE(AVERAGE('Database'[TP_CAL]),'Database'[TYPE] = "HTTP Download",'Database'[operator1] = "TRUE-H",'Database'[STATUS] = "PASS")
	RETURN
	SWITCH(
            SELECTEDVALUE('QoS_Parameter'[Index]),
            6, FORMAT(NoS_Ping, "#,##0"),
            17, FORMAT(NoS_HTTP, "#,##0"),
            27, FORMAT(NoS_FTPUL, "#,##0"),
            36, FORMAT(NoS_FTPDL, "#,##0"),
            5, FORMAT(PASS_ping, "#,##0"),
            11, FORMAT(PASS_http, "#,##0"),
            20, FORMAT(PASS_ftpul, "#,##0"),
            29, FORMAT(PASS_ftpdl, "#,##0"),
            13, FORMAT(FAIL_http, "#,##0"),
            22, FORMAT(FAIL_ftpul, "#,##0"),
            31, FORMAT(FAIL_ftpdl, "#,##0"),
            2, FORMAT(TP_PASS_ping,"#,##0"),
            15, FORMAT(TP_PASS_http, "#,##0"),
            24, FORMAT(TP_PASS_ftpul,"#,##0"),
            33, FORMAT(TP_PASS_ftpdl,"#,##0"),
            12, IF(ISBLANK(NoS_HTTP),"N/A",FORMAT(100 *
DIVIDE(PASS_http,NoS_HTTP),"##0.00")),
            21, IF(ISBLANK(NoS_FTPUL),"N/A",FORMAT(100 *
DIVIDE(PASS_ftpul,NoS_FTPUL),"##0.00")),
            30, IF(ISBLANK(NoS_FTPDL),"N/A",FORMAT(100*
DIVIDE(PASS_ftpdl,NoS_FTPDL),"##0.00")),
            14, IF(ISBLANK(NoS_HTTP),"N/A",FORMAT(100*
DIVIDE(FAIL_http,NoS_HTTP),"##0.00")),
            23, IF(ISBLANK(NoS_FTPUL),"N/A",FORMAT(100*
DIVIDE(FAIL_ftpul,NoS_FTPUL),"##0.00")),
            32, IF(ISBLANK(NoS_FTPDL),"N/A",FORMAT(100*
DIVIDE(FAIL_ftpdl,NoS_FTPDL),"##0.00")),
            4, IF(ISBLANK(PASS_ping),"N/A",FORMAT(100*
DIVIDE(TP_PASS_ping,PASS_ping),"##0.00")),
          26, IF(ISBLANK(PASS_ftpul),"N/A",FORMAT(100*
DIVIDE(TP_PASS_ftpul,PASS_ftpul),"##0.00")),
          35, IF(ISBLANK(PASS_ftpdl),"N/A",FORMAT(100*
DIVIDE(TP_PASS_ftpdl,PASS_ftpdl),"##0.00")),
          3, FORMAT(Avg_TP_Ping,"##0"),
          16, FORMAT(Avg_TP_HTTP / 1000,"#,##0.000"),
          25, FORMAT(Avg_TP_ftpul / 1000,"#,##0.000"),
          34, FORMAT(Avg_TP_ftpdl / 1000 ,"#,##0.000"),
          7, FORMAT(FAIL_ping,"#,##0"),
          8, IF(ISBLANK(NoS_Ping),"N/A",FORMAT(100 *
DIVIDE(FAIL_ping,NoS_Ping),"##0.00")),
          9, IF(ISBLANK(NoS_Ping),"N/A",FORMAT(100 *
DIVIDE(PASS_ping,NoS_Ping),"##0.00")),
          18, FORMAT(Avg_TP_web / 1000 ,"#,##0.000"),
          BLANK()
        )
```

<u>ส่วนของการแสดงผลของการบริการประเภทข้อมูล</u>

```
        CF_AWN =
        // ping
        VAR PASS_ping = CALCULATE(COUNTROWS('Database'), 'Database'[TYPE] =
"Ping", 'Database'[STATUS] = "PASS", 'Database'[operator1] = "TH GSM")
        VAR TP_PASS_ping = CALCULATE(COUNTROWS('Database'), 'Database'[TYPE]
= "Ping", 'Database'[STATUS] = "PASS", 'Database'[operator1] = "TH GSM",'Database'[TP2]
= "yes")
        // UL
        VAR NoS_FTPUL = CALCULATE(COUNTROWS('Database'),
'Database'[operator1] = "TH GSM",'Database'[TYPE] = "FTP Upload")
        VAR PASS_ftpul = CALCULATE(COUNTROWS('Database'),
'Database'[operator1] = "TH GSM",'Database'[TYPE] = "FTP Upload",'Database'[STATUS] =
"PASS")
```

```
        VAR TP_PASS_ftpul = CALCULATE(COUNTROWS('Database'),
'Database'[operator1] = "TH GSM",'Database'[TYPE] = "FTP Upload",'Database'[STATUS] =
"PASS",'Database'[TP2] = "yes")
        // DL
        VAR NoS_FTPDL = CALCULATE(COUNTROWS('Database'),
'Database'[operator1] = "TH GSM",'Database'[TYPE] = "FTP Download")
        VAR PASS_ftpdl = CALCULATE(COUNTROWS('Database'),
'Database'[operator1] = "TH GSM",'Database'[TYPE] = "FTP
Download",'Database'[STATUS] = "PASS")
        VAR TP_PASS_ftpdl = CALCULATE(COUNTROWS('Database'),
'Database'[operator1] = "TH GSM",'Database'[TYPE] = "FTP
Download",'Database'[STATUS] = "PASS",'Database'[TP2] = "yes")
        RETURN
        SWITCH(
          SELECTEDVALUE('QoS_Parameter'[Index]),
          // PING
          4, IF(DIVIDE(TP_PASS_ping,PASS_ping) >= 0.8 ,"#C1E1C1",
IF(DIVIDE(TP_PASS_ping,PASS_ping) < 0.8 ,"#FFB6B6")),
          // UL
          21, IF(DIVIDE(PASS_ftpul,NoS_FTPUL) >= 0.8 ,"#C1E1C1",
IF(DIVIDE(PASS_ftpul,NoS_FTPUL) < 0.8 ,"#FFB6B6")),
          26, IF(DIVIDE(TP_PASS_ftpul,PASS_ftpul) >= 0.8 ,"#C1E1C1",
IF(DIVIDE(TP_PASS_ftpul,PASS_ftpul) < 0.8 ,"#FFB6B6")),
          // DL
          30, IF(DIVIDE(PASS_ftpdl,NoS_FTPDL) >= 0.08 ,"#C1E1C1",
IF(DIVIDE(PASS_ftpdl,NoS_FTPDL) < 0.8 ,"#FFB6B6")),
          35, IF(DIVIDE(TP_PASS_ftpdl,PASS_ftpdl) >= 0.08 ,"#C1E1C1",
IF(DIVIDE(TP_PASS_ftpdl,PASS_ftpdl) < 0.8 ,"#FFB6B6"))
        )

        CF_CAT =
        // ping
        VAR PASS_ping = CALCULATE(COUNTROWS('Database'), 'Database'[TYPE] =
"Ping", 'Database'[STATUS] = "PASS", 'Database'[operator1] = "my")
        VAR TP_PASS_ping = CALCULATE(COUNTROWS('Database'), 'Database'[TYPE]
= "Ping", 'Database'[STATUS] = "PASS", 'Database'[operator1] = "my",'Database'[TP2] =
"yes")
        // UL
        VAR NoS_FTPUL = CALCULATE(COUNTROWS('Database'),
'Database'[operator1] = "my",'Database'[TYPE] = "FTP Upload")
        VAR PASS_ftpul = CALCULATE(COUNTROWS('Database'),
'Database'[operator1] = "my",'Database'[TYPE] = "FTP Upload",'Database'[STATUS] =
"PASS")
        VAR TP_PASS_ftpul = CALCULATE(COUNTROWS('Database'),
'Database'[operator1] = "my",'Database'[TYPE] = "FTP Upload",'Database'[STATUS] =
"PASS",'Database'[TP2] = "yes")
        // DL
        VAR NoS_FTPDL = CALCULATE(COUNTROWS('Database'),
'Database'[operator1] = "my",'Database'[TYPE] = "FTP Download")
        VAR PASS_ftpdl = CALCULATE(COUNTROWS('Database'),
'Database'[operator1] = "my",'Database'[TYPE] = "FTP Download",'Database'[STATUS] =
"PASS")
        VAR TP_PASS_ftpdl = CALCULATE(COUNTROWS('Database'),
'Database'[operator1] = "my",'Database'[TYPE] = "FTP Download",'Database'[STATUS] =
"PASS",'Database'[TP2] = "yes")
        RETURN
        SWITCH(
          SELECTEDVALUE('QoS_Parameter'[Index]),
          // PING
          4, IF(DIVIDE(TP_PASS_ping,PASS_ping) >= 0.8 ,"#C1E1C1",
IF(DIVIDE(TP_PASS_ping,PASS_ping) < 0.8 ,"#FFB6B6")),
          // UL
          21, IF(DIVIDE(PASS_ftpul,NoS_FTPUL) >= 0.8 ,"#C1E1C1",
IF(DIVIDE(PASS_ftpul,NoS_FTPUL) < 0.8 ,"#FFB6B6")),
          26, IF(DIVIDE(TP_PASS_ftpul,PASS_ftpul) >= 0.8 ,"#C1E1C1",
IF(DIVIDE(TP_PASS_ftpul,PASS_ftpul) < 0.8 ,"#FFB6B6")),
          // DL
          30, IF(DIVIDE(PASS_ftpdl,NoS_FTPDL) >= 0.08 ,"#C1E1C1",
IF(DIVIDE(PASS_ftpdl,NoS_FTPDL) < 0.8 ,"#FFB6B6")),
          35, IF(DIVIDE(TP_PASS_ftpdl,PASS_ftpdl) >= 0.08 ,"#C1E1C1",
IF(DIVIDE(TP_PASS_ftpdl,PASS_ftpdl) < 0.8 ,"#FFB6B6"))
        )


        CF_DTN =
        // ping
        VAR PASS_ping = CALCULATE(COUNTROWS('Database'), 'Database'[TYPE] =
"Ping", 'Database'[STATUS] = "PASS", 'Database'[operator1] = "DTAC")
        VAR TP_PASS_ping = CALCULATE(COUNTROWS('Database'), 'Database'[TYPE]
= "Ping", 'Database'[STATUS] = "PASS", 'Database'[operator1] = "DTAC",'Database'[TP2] =
"yes")
        // UL
        VAR NoS_FTPUL = CALCULATE(COUNTROWS('Database'),
'Database'[operator1] = "DTAC",'Database'[TYPE] = "FTP Upload")
        VAR PASS_ftpul = CALCULATE(COUNTROWS('Database'),
'Database'[operator1] = "DTAC",'Database'[TYPE] = "FTP Upload",'Database'[STATUS] =
"PASS")
        VAR TP_PASS_ftpul = CALCULATE(COUNTROWS('Database'),
'Database'[operator1] = "DTAC",'Database'[TYPE] = "FTP Upload",'Database'[STATUS] =
"PASS",'Database'[TP2] = "yes")
        // DL
        VAR NoS_FTPDL = CALCULATE(COUNTROWS('Database'),
'Database'[operator1] = "DTAC",'Database'[TYPE] = "FTP Download")
```

```
        VAR PASS_ftpdl = CALCULATE(COUNTROWS('Database'),
'Database'[operator1] = "DTAC",'Database'[TYPE] = "FTP Download",'Database'[STATUS] =
"PASS")
        VAR TP_PASS_ftpdl = CALCULATE(COUNTROWS('Database'),
'Database'[operator1] = "DTAC",'Database'[TYPE] = "FTP Download",'Database'[STATUS] =
"PASS",'Database'[TP2] = "yes")
        RETURN
        SWITCH(
          SELECTEDVALUE('QoS_Parameter'[Index]),
          // PING
          4, IF(DIVIDE(TP_PASS_ping,PASS_ping) >= 0.8 ,"#C1E1C1",
IF(DIVIDE(TP_PASS_ping,PASS_ping) < 0.8 ,"#FFB6B6")),
          // UL
          21, IF(DIVIDE(PASS_ftpul,NoS_FTPUL) >= 0.8 ,"#C1E1C1",
IF(DIVIDE(PASS_ftpul,NoS_FTPUL) < 0.8 ,"#FFB6B6")),
          26, IF(DIVIDE(TP_PASS_ftpul,PASS_ftpul) >= 0.8 ,"#C1E1C1",
IF(DIVIDE(TP_PASS_ftpul,PASS_ftpul) < 0.8 ,"#FFB6B6")),
          // DL
          30, IF(DIVIDE(PASS_ftpdl,NoS_FTPDL) >= 0.08 ,"#C1E1C1",
IF(DIVIDE(PASS_ftpdl,NoS_FTPDL) < 0.8 ,"#FFB6B6")),
          35, IF(DIVIDE(TP_PASS_ftpdl,PASS_ftpdl) >= 0.08 ,"#C1E1C1",
IF(DIVIDE(TP_PASS_ftpdl,PASS_ftpdl) < 0.8 ,"#FFB6B6"))
        )

        CF_TOT =
        // ping
        VAR PASS_ping = CALCULATE(COUNTROWS('Database'), 'Database'[TYPE] =
"Ping", 'Database'[STATUS] = "PASS", 'Database'[operator1] = "TOT")
        VAR TP_PASS_ping = CALCULATE(COUNTROWS('Database'), 'Database'[TYPE]
= "Ping", 'Database'[STATUS] = "PASS", 'Database'[operator1] = "TOT",'Database'[TP2] =
"yes")
```

```
        // UL
        VAR NoS_FTPUL = CALCULATE(COUNTROWS('Database'),
'Database'[operator1] = "TOT",'Database'[TYPE] = "FTP Upload")
        VAR PASS_ftpul = CALCULATE(COUNTROWS('Database'),
'Database'[operator1] = "TOT",'Database'[TYPE] = "FTP Upload",'Database'[STATUS] =
"PASS")
        VAR TP_PASS_ftpul = CALCULATE(COUNTROWS('Database'),
'Database'[operator1] = "TOT",'Database'[TYPE] = "FTP Upload",'Database'[STATUS] =
"PASS",'Database'[TP2] = "yes")
        // DL
        VAR NoS_FTPDL = CALCULATE(COUNTROWS('Database'),
'Database'[operator1] = "TOT",'Database'[TYPE] = "FTP Download")
        VAR PASS_ftpdl = CALCULATE(COUNTROWS('Database'),
'Database'[operator1] = "TOT",'Database'[TYPE] = "FTP Download",'Database'[STATUS] =
"PASS")
        VAR TP_PASS_ftpdl = CALCULATE(COUNTROWS('Database'),
'Database'[operator1] = "TOT",'Database'[TYPE] = "FTP Download",'Database'[STATUS] =
"PASS",'Database'[TP2] = "yes")
        RETURN
        SWITCH(
          SELECTEDVALUE('QoS_Parameter'[Index]),
          // PING
          4, IF(DIVIDE(TP_PASS_ping,PASS_ping) >= 0.8 ,"#C1E1C1",
IF(DIVIDE(TP_PASS_ping,PASS_ping) < 0.8 ,"#FFB6B6")),
          // UL
          21, IF(DIVIDE(PASS_ftpul,NoS_FTPUL) >= 0.8 ,"#C1E1C1",
IF(DIVIDE(PASS_ftpul,NoS_FTPUL) < 0.8 ,"#FFB6B6")),
          26, IF(DIVIDE(TP_PASS_ftpul,PASS_ftpul) >= 0.8 ,"#C1E1C1",
IF(DIVIDE(TP_PASS_ftpul,PASS_ftpul) < 0.8 ,"#FFB6B6")),
          // DL
```

```
            30, IF(DIVIDE(PASS_ftpdl,NoS_FTPDL) >= 0.08 ,"#C1E1C1", IF(DIVIDE(PASS_ftpdl,NoS_FTPDL) < 0.8 ,"#FFB6B6")),
            35, IF(DIVIDE(TP_PASS_ftpdl,PASS_ftpdl) >= 0.08 ,"#C1E1C1", IF(DIVIDE(TP_PASS_ftpdl,PASS_ftpdl) < 0.8 ,"#FFB6B6"))
        )

        CF_TRUE =
        // ping
        VAR PASS_ping = CALCULATE(COUNTROWS('Database'), 'Database'[TYPE] = "Ping", 'Database'[STATUS] = "PASS", 'Database'[operator1] = "TRUE-H")
        VAR TP_PASS_ping = CALCULATE(COUNTROWS('Database'), 'Database'[TYPE] = "Ping", 'Database'[STATUS] = "PASS", 'Database'[operator1] = "TRUE-H",'Database'[TP2] = "yes")
        // UL
        VAR NoS_FTPUL = CALCULATE(COUNTROWS('Database'), 'Database'[operator1] = "TRUE-H",'Database'[TYPE] = "FTP Upload")
        VAR PASS_ftpul = CALCULATE(COUNTROWS('Database'), 'Database'[operator1] = "TRUE-H",'Database'[TYPE] = "FTP Upload",'Database'[STATUS] = "PASS")
        VAR TP_PASS_ftpul = CALCULATE(COUNTROWS('Database'), 'Database'[operator1] = "TRUE-H",'Database'[TYPE] = "FTP Upload",'Database'[STATUS] = "PASS",'Database'[TP2] = "yes")
        // DL
        VAR NoS_FTPDL = CALCULATE(COUNTROWS('Database'), 'Database'[operator1] = "TRUE-H",'Database'[TYPE] = "FTP Download")
        VAR PASS_ftpdl = CALCULATE(COUNTROWS('Database'), 'Database'[operator1] = "TRUE-H",'Database'[TYPE] = "FTP Download",'Database'[STATUS] = "PASS")
        VAR TP_PASS_ftpdl = CALCULATE(COUNTROWS('Database'), 'Database'[operator1] = "TRUE-H",'Database'[TYPE] = "FTP Download",'Database'[STATUS] = "PASS",'Database'[TP2] = "yes")
```

```
RETURN
SWITCH(
    SELECTEDVALUE('QoS_Parameter'[Index]),
    // PING
    4, IF(DIVIDE(TP_PASS_ping,PASS_ping) >= 0.8 ,"#C1E1C1",
IF(DIVIDE(TP_PASS_ping,PASS_ping) < 0.8 ,"#FFB6B6")),
    // UL
    21, IF(DIVIDE(PASS_ftpul,NoS_FTPUL) >= 0.8 ,"#C1E1C1",
IF(DIVIDE(PASS_ftpul,NoS_FTPUL) < 0.8 ,"#FFB6B6")),
    26, IF(DIVIDE(TP_PASS_ftpul,PASS_ftpul) >= 0.8 ,"#C1E1C1",
IF(DIVIDE(TP_PASS_ftpul,PASS_ftpul) < 0.8 ,"#FFB6B6")),
    // DL
    30, IF(DIVIDE(PASS_ftpdl,NoS_FTPDL) >= 0.08 ,"#C1E1C1",
IF(DIVIDE(PASS_ftpdl,NoS_FTPDL) < 0.8 ,"#FFB6B6")),
    35, IF(DIVIDE(TP_PASS_ftpdl,PASS_ftpdl) >= 0.08 ,"#C1E1C1",
IF(DIVIDE(TP_PASS_ftpdl,PASS_ftpdl) < 0.8 ,"#FFB6B6"))
)
```

<u>ส่วนของการคำนวณของการบริการประเภทเสียง</u>

```
AWN_voc =
VAR VOICE_SAMPLE = CALCULATE(COUNTAX(FILTER('voice database','voice
database'[TYPE] = "Voice Successful" && 'voice database'[operator1] = "TH GSM" &&
'voice database'[TP] in {"Complete","Drop","Block"}),'voice database'[IMEI]))
VAR VOICE_COMPLETE = CALCULATE(COUNTAX(FILTER('voice
database','voice database'[TYPE] = "Voice Successful" && 'voice database'[operator1] =
"TH GSM" && 'voice database'[TP] = "Complete"),'voice database'[IMEI]))
VAR VOICE_BLOCK = CALCULATE(COUNTAX(FILTER('voice database','voice
database'[TYPE] = "Voice Successful" && 'voice database'[operator1] = "TH GSM" &&
'voice database'[TP] = "Block"),'voice database'[IMEI]))
```

```
VAR VOICE_DROP = CALCULATE(COUNTAX(FILTER('voice database','voice database'[TYPE] = "Voice Successful" && 'voice database'[operator1] = "TH GSM" && 'voice database'[TP] = "Drop"),'voice database'[IMEI]))
VAR VOICE_BAD = CALCULATE(COUNTAX(FILTER('voice database','voice database'[TYPE] = "Voice Successful" && 'voice database'[operator1] = "TH GSM" && 'voice database'[TP] = "Bad Attempt"),'voice database'[IMEI]))
RETURN
SWITCH(
    SELECTEDVALUE('QoS_Parameter(Voice)'[Index]),
    1, BLANK(),
    2, IF(ISBLANK(VOICE_SAMPLE), BLANK(), FORMAT(VOICE_SAMPLE, "#,##0")),
    3, IF(ISBLANK(VOICE_COMPLETE), BLANK(), FORMAT(VOICE_COMPLETE, "#,##0")),
    4, IF(ISBLANK(VOICE_SAMPLE) || ISBLANK(VOICE_COMPLETE + VOICE_DROP), "0.00%", FORMAT(DIVIDE(VOICE_COMPLETE + VOICE_DROP, VOICE_SAMPLE), "0.00%")),
    5, IF(ISBLANK(VOICE_BLOCK), BLANK(), FORMAT(VOICE_BLOCK, "#,##0")),
    6, IF(ISBLANK(VOICE_DROP), BLANK(), FORMAT(VOICE_DROP, "#,##0")),
    7, IF(ISBLANK(VOICE_DROP), BLANK(), FORMAT(DIVIDE(VOICE_DROP, VOICE_SAMPLE - VOICE_BLOCK), "0.00%")),
    8, IF(ISBLANK(VOICE_COMPLETE + VOICE_DROP), BLANK(), FORMAT(VOICE_COMPLETE + VOICE_DROP, "#,##0")),
    9, IF(ISBLANK(VOICE_SAMPLE) || ISBLANK(VOICE_COMPLETE), "0.00%", FORMAT(DIVIDE(VOICE_COMPLETE, VOICE_SAMPLE), "0.00%")),
    10, IF(ISBLANK(VOICE_BAD), BLANK(), FORMAT(VOICE_BAD, "#,##0")),
    11, IF(ISBLANK(VOICE_SAMPLE) || ISBLANK(VOICE_BAD), "0.00%", FORMAT(DIVIDE(VOICE_BAD, VOICE_SAMPLE), "0.00%")),
    0
)
```

```
CAT_voc =
VAR VOICE_SAMPLE = CALCULATE(COUNTAX(FILTER('voice database','voice database'[TYPE] = "Voice Successful" && 'voice database'[operator1] = "my" && 'voice database'[TP] in {"Complete","Drop","Block"}),'voice database'[IMEI]))
VAR VOICE_COMPLETE = CALCULATE(COUNTAX(FILTER('voice database','voice database'[TYPE] = "Voice Successful" && 'voice database'[operator1] = "my" && 'voice database'[TP] = "Complete"),'voice database'[IMEI]))
VAR VOICE_BLOCK = CALCULATE(COUNTAX(FILTER('voice database','voice database'[TYPE] = "Voice Successful" && 'voice database'[operator1] = "my" && 'voice database'[TP] = "Block"),'voice database'[IMEI]))
VAR VOICE_DROP = CALCULATE(COUNTAX(FILTER('voice database','voice database'[TYPE] = "Voice Successful" && 'voice database'[operator1] = "my" && 'voice database'[TP] = "Drop"),'voice database'[IMEI]))
VAR VOICE_BAD = CALCULATE(COUNTAX(FILTER('voice database','voice database'[TYPE] = "Voice Successful" && 'voice database'[operator1] = "my" && 'voice database'[TP] = "Bad Attempt"),'voice database'[IMEI]))
RETURN
SWITCH(
    SELECTEDVALUE('QoS_Parameter(Voice)'[Index]),
    1, BLANK(),
    2, IF(ISBLANK(VOICE_SAMPLE), BLANK(), FORMAT(VOICE_SAMPLE, "#,##0")),
    3, IF(ISBLANK(VOICE_COMPLETE), BLANK(), FORMAT(VOICE_COMPLETE, "#,##0")),
    4, IF(ISBLANK(VOICE_SAMPLE) || ISBLANK(VOICE_COMPLETE + VOICE_DROP), "0.00%", FORMAT(DIVIDE(VOICE_COMPLETE + VOICE_DROP, VOICE_SAMPLE), "0.00%")),
    5, IF(ISBLANK(VOICE_BLOCK), BLANK(), FORMAT(VOICE_BLOCK, "#,##0")),
    6, IF(ISBLANK(VOICE_DROP), BLANK(), FORMAT(VOICE_DROP, "#,##0")),
    7, IF(ISBLANK(VOICE_DROP), BLANK(), FORMAT(DIVIDE(VOICE_DROP, VOICE_SAMPLE - VOICE_BLOCK), "0.00%")),
        8, IF(ISBLANK(VOICE_COMPLETE + VOICE_DROP), BLANK(),
FORMAT(VOICE_COMPLETE + VOICE_DROP, "#,##0")),
        9, IF(ISBLANK(VOICE_SAMPLE) || ISBLANK(VOICE_COMPLETE), "0.00%",
FORMAT(DIVIDE(VOICE_COMPLETE, VOICE_SAMPLE), "0.00%")),
        10, IF(ISBLANK(VOICE_BAD), BLANK(), FORMAT(VOICE_BAD, "#,##0")),
        11, IF(ISBLANK(VOICE_SAMPLE) || ISBLANK(VOICE_BAD), "0.00%",
FORMAT(DIVIDE(VOICE_BAD, VOICE_SAMPLE), "0.00%")),
        0
    )


    DTN_voc =
    VAR VOICE_SAMPLE = CALCULATE(COUNTAX(FILTER('voice database','voice
database'[TYPE] = "Voice Successful" && 'voice database'[operator1] = "DTAC" && 'voice
database'[TP] in {"Complete","Drop","Block"}),'voice database'[IMEI]))
    VAR VOICE_COMPLETE = CALCULATE(COUNTAX(FILTER('voice
database','voice database'[TYPE] = "Voice Successful" && 'voice database'[operator1] =
"DTAC" && 'voice database'[TP] = "Complete"),'voice database'[IMEI]))
    VAR VOICE_BLOCK = CALCULATE(COUNTAX(FILTER('voice database','voice
database'[TYPE] = "Voice Successful" && 'voice database'[operator1] = "DTAC" && 'voice
database'[TP] = "Block"),'voice database'[IMEI]))
    VAR VOICE_DROP = CALCULATE(COUNTAX(FILTER('voice database','voice
database'[TYPE] = "Voice Successful" && 'voice database'[operator1] = "DTAC" && 'voice
database'[TP] = "Drop"),'voice database'[IMEI]))
    VAR VOICE_BAD = CALCULATE(COUNTAX(FILTER('voice database','voice
database'[TYPE] = "Voice Successful" && 'voice database'[operator1] = "DTAC" && 'voice
database'[TP] = "Bad Attempt"),'voice database'[IMEI]))
    RETURN
    SWITCH(
        SELECTEDVALUE('QoS_Parameter(Voice)'[Index]),
        1, BLANK(),
        2, IF(ISBLANK(VOICE_SAMPLE), BLANK(), FORMAT(VOICE_SAMPLE, "#,##0")),
        3, IF(ISBLANK(VOICE_COMPLETE), BLANK(), FORMAT(VOICE_COMPLETE, "#,##0")),
        4, IF(ISBLANK(VOICE_SAMPLE) || ISBLANK(VOICE_COMPLETE + VOICE_DROP), "0.00%", FORMAT(DIVIDE(VOICE_COMPLETE + VOICE_DROP, VOICE_SAMPLE), "0.00%")),
        5, IF(ISBLANK(VOICE_BLOCK), BLANK(), FORMAT(VOICE_BLOCK, "#,##0")),
        6, IF(ISBLANK(VOICE_DROP), BLANK(), FORMAT(VOICE_DROP, "#,##0")),
        7, IF(ISBLANK(VOICE_DROP), BLANK(), FORMAT(DIVIDE(VOICE_DROP, VOICE_SAMPLE - VOICE_BLOCK), "0.00%")),
        8, IF(ISBLANK(VOICE_COMPLETE + VOICE_DROP), BLANK(), FORMAT(VOICE_COMPLETE + VOICE_DROP, "#,##0")),
        9, IF(ISBLANK(VOICE_SAMPLE) || ISBLANK(VOICE_COMPLETE), "0.00%", FORMAT(DIVIDE(VOICE_COMPLETE, VOICE_SAMPLE), "0.00%")),
        10, IF(ISBLANK(VOICE_BAD), BLANK(), FORMAT(VOICE_BAD, "#,##0")),
        11, IF(ISBLANK(VOICE_SAMPLE) || ISBLANK(VOICE_BAD), "0.00%", FORMAT(DIVIDE(VOICE_BAD, VOICE_SAMPLE), "0.00%")),
        0
    )


    TOT_voc =
    VAR VOICE_SAMPLE = CALCULATE(COUNTAX(FILTER('voice database','voice database'[TYPE] = "Voice Successful" && 'voice database'[operator1] = "TOT" && 'voice database'[TP] in {"Complete","Drop","Block"}),'voice database'[IMEI]))
    VAR VOICE_COMPLETE = CALCULATE(COUNTAX(FILTER('voice database','voice database'[TYPE] = "Voice Successful" && 'voice database'[operator1] = "TOT" && 'voice database'[TP] = "Complete"),'voice database'[IMEI]))
    VAR VOICE_BLOCK = CALCULATE(COUNTAX(FILTER('voice database','voice database'[TYPE] = "Voice Successful" && 'voice database'[operator1] = "TOT" && 'voice database'[TP] = "Block"),'voice database'[IMEI]))
```

```
        VAR VOICE_DROP = CALCULATE(COUNTAX(FILTER('voice database','voice
database'[TYPE] = "Voice Successful" && 'voice database'[operator1] = "TOT" && 'voice
database'[TP] = "Drop"),'voice database'[IMEI]))
        VAR VOICE_BAD = CALCULATE(COUNTAX(FILTER('voice database','voice
database'[TYPE] = "Voice Successful" && 'voice database'[operator1] = "TOT" && 'voice
database'[TP] = "Bad Attempt"),'voice database'[IMEI]))
        RETURN
        SWITCH(
          SELECTEDVALUE('QoS_Parameter(Voice)'[Index]),
          1, BLANK(),
          2, IF(ISBLANK(VOICE_SAMPLE), BLANK(), FORMAT(VOICE_SAMPLE,
"#,##0")),
          3, IF(ISBLANK(VOICE_COMPLETE), BLANK(), FORMAT(VOICE_COMPLETE,
"#,##0")),
          4, IF(ISBLANK(VOICE_SAMPLE) || ISBLANK(VOICE_COMPLETE +
VOICE_DROP), "0.00%", FORMAT(DIVIDE(VOICE_COMPLETE + VOICE_DROP,
VOICE_SAMPLE), "0.00%")),
          5, IF(ISBLANK(VOICE_BLOCK), BLANK(), FORMAT(VOICE_BLOCK, "#,##0")),
          6, IF(ISBLANK(VOICE_DROP), BLANK(), FORMAT(VOICE_DROP, "#,##0")),
          7, IF(ISBLANK(VOICE_DROP), BLANK(), FORMAT(DIVIDE(VOICE_DROP,
VOICE_SAMPLE - VOICE_BLOCK), "0.00%")),
          8, IF(ISBLANK(VOICE_COMPLETE + VOICE_DROP), BLANK(),
FORMAT(VOICE_COMPLETE + VOICE_DROP, "#,##0")),
          9, IF(ISBLANK(VOICE_SAMPLE) || ISBLANK(VOICE_COMPLETE), "0.00%",
FORMAT(DIVIDE(VOICE_COMPLETE, VOICE_SAMPLE), "0.00%")),
          10, IF(ISBLANK(VOICE_BAD), BLANK(), FORMAT(VOICE_BAD, "#,##0")),
          11, IF(ISBLANK(VOICE_SAMPLE) || ISBLANK(VOICE_BAD), "0.00%",
FORMAT(DIVIDE(VOICE_BAD, VOICE_SAMPLE), "0.00%")),
          0
        )
```

```
TUC_voc =
VAR VOICE_SAMPLE = CALCULATE(COUNTAX(FILTER('voice database','voice database'[TYPE] = "Voice Successful" && 'voice database'[operator1] = "TRUE-H" && 'voice database'[TP] in {"Complete","Drop","Block"}),'voice database'[IMEI]))
VAR VOICE_COMPLETE = CALCULATE(COUNTAX(FILTER('voice database','voice database'[TYPE] = "Voice Successful" && 'voice database'[operator1] = "TRUE-H" && 'voice database'[TP] = "Complete"),'voice database'[IMEI]))
VAR VOICE_BLOCK = CALCULATE(COUNTAX(FILTER('voice database','voice database'[TYPE] = "Voice Successful" && 'voice database'[operator1] = "TRUE-H" && 'voice database'[TP] = "Block"),'voice database'[IMEI]))
VAR VOICE_DROP = CALCULATE(COUNTAX(FILTER('voice database','voice database'[TYPE] = "Voice Successful" && 'voice database'[operator1] = "TRUE-H" && 'voice database'[TP] = "Drop"),'voice database'[IMEI]))
VAR VOICE_BAD = CALCULATE(COUNTAX(FILTER('voice database','voice database'[TYPE] = "Voice Successful" && 'voice database'[operator1] = "TRUE-H" && 'voice database'[TP] = "Bad Attempt"),'voice database'[IMEI]))
RETURN
SWITCH(
    SELECTEDVALUE('QoS_Parameter(Voice)'[Index]),
    1, BLANK(),
    2, IF(ISBLANK(VOICE_SAMPLE), BLANK(), FORMAT(VOICE_SAMPLE, "#,##0")),
    3, IF(ISBLANK(VOICE_COMPLETE), BLANK(), FORMAT(VOICE_COMPLETE, "#,##0")),
    4, IF(ISBLANK(VOICE_SAMPLE) || ISBLANK(VOICE_COMPLETE + VOICE_DROP), "0.00%", FORMAT(DIVIDE(VOICE_COMPLETE + VOICE_DROP, VOICE_SAMPLE), "0.00%")),
    5, IF(ISBLANK(VOICE_BLOCK), BLANK(), FORMAT(VOICE_BLOCK, "#,##0")),
    6, IF(ISBLANK(VOICE_DROP), BLANK(), FORMAT(VOICE_DROP, "#,##0")),
    7, IF(ISBLANK(VOICE_DROP), BLANK(), FORMAT(DIVIDE(VOICE_DROP, VOICE_SAMPLE - VOICE_BLOCK), "0.00%")),
            8, IF(ISBLANK(VOICE_COMPLETE + VOICE_DROP), BLANK(),
FORMAT(VOICE_COMPLETE + VOICE_DROP, "#,##0")),
            9, IF(ISBLANK(VOICE_SAMPLE) || ISBLANK(VOICE_COMPLETE), "0.00%",
FORMAT(DIVIDE(VOICE_COMPLETE, VOICE_SAMPLE), "0.00%")),
            10, IF(ISBLANK(VOICE_BAD), BLANK(), FORMAT(VOICE_BAD, "#,##0")),
            11, IF(ISBLANK(VOICE_SAMPLE) || ISBLANK(VOICE_BAD), "0.00%",
FORMAT(DIVIDE(VOICE_BAD, VOICE_SAMPLE), "0.00%")),
            0
        )
```

ส่วนของการแสดงผลของการบริการประเภทเสียง

```
        CF_AWN_voc =
        VAR VOICE_SAMPLE = CALCULATE(COUNTROWS('Database'),
'Database'[TYPE] = "Voice Successful", 'Database'[operator1] = "TH GSM")
        VAR VOICE_COMPLETE = CALCULATE(COUNTROWS('Database'),
'Database'[TYPE] = "Voice Successful", 'Database'[operator1] = "TH GSM", 'Database'[TP]
= "Complete")
        VAR VOICE_BLOCK = CALCULATE(COUNTROWS('Database'), 'Database'[TYPE]
= "Voice Successful", 'Database'[operator1] = "TH GSM", 'Database'[TP] = "Block")
        VAR VOICE_DROP = CALCULATE(COUNTROWS('Database'), 'Database'[TYPE]
= "Voice Successful", 'Database'[operator1] = "TH GSM", 'Database'[TP] = "Drop")
        VAR VOICE_BAD = CALCULATE(COUNTROWS('Database'), 'Database'[TYPE] =
"Voice Successful", 'Database'[operator1] = "TH GSM", 'Database'[TP] = "Bad")
        RETURN
        SWITCH(
            SELECTEDVALUE('QoS_Parameter(Voice)'[Index]),
            // ให้ success call setup >= 90%
            4, IF(DIVIDE(VOICE_COMPLETE + VOICE_DROP, VOICE_SAMPLE) >= 0.9 &&
DIVIDE(VOICE_COMPLETE + VOICE_DROP, VOICE_SAMPLE) <= 1.0, "#C1E1C1",
           IF(DIVIDE(VOICE_COMPLETE + VOICE_DROP, VOICE_SAMPLE) < 0.9,
"#FFB6B6", BLANK())),
           // drop call rate <= 2%
           7, IF(DIVIDE(VOICE_DROP, VOICE_SAMPLE - VOICE_BLOCK) >= 0.9 && DIVIDE(VOICE_DROP, VOICE_SAMPLE - VOICE_BLOCK) <= 1.0, "#FFB6B6",
           IF(DIVIDE(VOICE_DROP, VOICE_SAMPLE - VOICE_BLOCK) < 0.9, "#C1E1C1", BLANK())),
           // successful call ratio >= 90%
           9, IF(DIVIDE(VOICE_COMPLETE, VOICE_SAMPLE) >= 0.9 && DIVIDE(VOICE_COMPLETE, VOICE_SAMPLE) <= 1.0, "#C1E1C1",
           IF(DIVIDE(VOICE_COMPLETE, VOICE_SAMPLE) < 0.9, "#FFB6B6", BLANK())),
           // ให้ bad call rate <= 2%
           11, IF(DIVIDE(VOICE_BAD, VOICE_SAMPLE) >= 0.9 && DIVIDE(VOICE_BAD, VOICE_SAMPLE) <= 1.0, "#FFB6B6",
           IF(DIVIDE(VOICE_BAD, VOICE_SAMPLE) < 0.9, "#C1E1C1", BLANK()))
        )


        CF_CAT_voc =
        VAR VOICE_SAMPLE = CALCULATE(COUNTROWS('Database'), 'Database'[TYPE] = "Voice Successful", 'Database'[operator1] = "my")
        VAR VOICE_COMPLETE = CALCULATE(COUNTROWS('Database'), 'Database'[TYPE] = "Voice Successful", 'Database'[operator1] = "my", 'Database'[TP] = "Complete")
        VAR VOICE_BLOCK = CALCULATE(COUNTROWS('Database'), 'Database'[TYPE] = "Voice Successful", 'Database'[operator1] = "my", 'Database'[TP] = "Block")
        VAR VOICE_DROP = CALCULATE(COUNTROWS('Database'), 'Database'[TYPE] = "Voice Successful", 'Database'[operator1] = "my", 'Database'[TP] = "Drop")
        VAR VOICE_BAD = CALCULATE(COUNTROWS('Database'), 'Database'[TYPE] = "Voice Successful", 'Database'[operator1] = "my", 'Database'[TP] = "Bad")
        RETURN
        SWITCH(
            SELECTEDVALUE('QoS_Parameter(Voice)'[Index]),
            // ให้ success call setup >= 90%
```

```
            4, IF(DIVIDE(VOICE_COMPLETE + VOICE_DROP, VOICE_SAMPLE) >= 0.9 &&
DIVIDE(VOICE_COMPLETE + VOICE_DROP, VOICE_SAMPLE) <= 1.0, "#C1E1C1",
            IF(DIVIDE(VOICE_COMPLETE + VOICE_DROP, VOICE_SAMPLE) < 0.9,
"#FFB6B6", BLANK())),
            // drop call rate <= 2%
            7, IF(DIVIDE(VOICE_DROP, VOICE_SAMPLE - VOICE_BLOCK) >= 0.9 &&
DIVIDE(VOICE_DROP, VOICE_SAMPLE - VOICE_BLOCK) <= 1.0, "#FFB6B6",
            IF(DIVIDE(VOICE_DROP, VOICE_SAMPLE - VOICE_BLOCK) < 0.9, "#C1E1C1",
BLANK())),
            // successful call ratio >= 90%
            9, IF(DIVIDE(VOICE_COMPLETE, VOICE_SAMPLE) >= 0.9 &&
DIVIDE(VOICE_COMPLETE, VOICE_SAMPLE) <= 1.0, "#C1E1C1",
            IF(DIVIDE(VOICE_COMPLETE, VOICE_SAMPLE) < 0.9, "#FFB6B6", BLANK())),
            // ให้ bad call rate <= 2%
            11, IF(DIVIDE(VOICE_BAD, VOICE_SAMPLE) >= 0.9 && DIVIDE(VOICE_BAD,
VOICE_SAMPLE) <= 1.0, "#FFB6B6",
            IF(DIVIDE(VOICE_BAD, VOICE_SAMPLE) < 0.9, "#C1E1C1", BLANK()))
          )

          CF_DTN_voc =
          VAR VOICE_SAMPLE = CALCULATE(COUNTROWS('Database'),
'Database'[TYPE] = "Voice Successful", 'Database'[operator1] = "DTAC")
          VAR VOICE_COMPLETE = CALCULATE(COUNTROWS('Database'),
'Database'[TYPE] = "Voice Successful", 'Database'[operator1] = "DTAC", 'Database'[TP] =
"Complete")
          VAR VOICE_BLOCK = CALCULATE(COUNTROWS('Database'), 'Database'[TYPE]
= "Voice Successful", 'Database'[operator1] = "DTAC", 'Database'[TP] = "Block")
          VAR VOICE_DROP = CALCULATE(COUNTROWS('Database'), 'Database'[TYPE]
= "Voice Successful", 'Database'[operator1] = "DTAC", 'Database'[TP] = "Drop")
          VAR VOICE_BAD = CALCULATE(COUNTROWS('Database'), 'Database'[TYPE] =
"Voice Successful", 'Database'[operator1] = "DTAC", 'Database'[TP] = "Bad")
```

```
        RETURN
        SWITCH(
          SELECTEDVALUE('QoS_Parameter(Voice)'[Index]),
          // ให้ success call setup >= 90%
          4, IF(DIVIDE(VOICE_COMPLETE + VOICE_DROP, VOICE_SAMPLE) >= 0.9 &&
DIVIDE(VOICE_COMPLETE + VOICE_DROP, VOICE_SAMPLE) <= 1.0, "#C1E1C1",
         IF(DIVIDE(VOICE_COMPLETE + VOICE_DROP, VOICE_SAMPLE) < 0.9,
"#FFB6B6", BLANK())),
         // drop call rate <= 2%
         7, IF(DIVIDE(VOICE_DROP, VOICE_SAMPLE - VOICE_BLOCK) >= 0.9 &&
DIVIDE(VOICE_DROP, VOICE_SAMPLE - VOICE_BLOCK) <= 1.0, "#FFB6B6",
         IF(DIVIDE(VOICE_DROP, VOICE_SAMPLE - VOICE_BLOCK) < 0.9, "#C1E1C1",
BLANK())),
         // successful call ratio >= 90%
         9, IF(DIVIDE(VOICE_COMPLETE, VOICE_SAMPLE) >= 0.9 &&
DIVIDE(VOICE_COMPLETE, VOICE_SAMPLE) <= 1.0, "#C1E1C1",
         IF(DIVIDE(VOICE_COMPLETE, VOICE_SAMPLE) < 0.9, "#FFB6B6", BLANK())),
         // ให้ bad call rate <= 2%
         11, IF(DIVIDE(VOICE_BAD, VOICE_SAMPLE) >= 0.9 && DIVIDE(VOICE_BAD,
VOICE_SAMPLE) <= 1.0, "#FFB6B6",
         IF(DIVIDE(VOICE_BAD, VOICE_SAMPLE) < 0.9, "#C1E1C1", BLANK()))
        )

        CF_TOT_voc =
        VAR VOICE_SAMPLE = CALCULATE(COUNTROWS('Database'),
'Database'[TYPE] = "Voice Successful", 'Database'[operator1] = "TOT")
        VAR VOICE_COMPLETE = CALCULATE(COUNTROWS('Database'),
'Database'[TYPE] = "Voice Successful", 'Database'[operator1] = "TOT", 'Database'[TP] =
"Complete")
        VAR VOICE_BLOCK = CALCULATE(COUNTROWS('Database'), 'Database'[TYPE]
= "Voice Successful", 'Database'[operator1] = "TOT", 'Database'[TP] = "Block")
```

```
VAR VOICE_DROP = CALCULATE(COUNTROWS('Database'), 'Database'[TYPE]
= "Voice Successful", 'Database'[operator1] = "TOT", 'Database'[TP] = "Drop")
VAR VOICE_BAD = CALCULATE(COUNTROWS('Database'), 'Database'[TYPE] =
"Voice Successful", 'Database'[operator1] = "TOT", 'Database'[TP] = "Bad")
RETURN
SWITCH(
    SELECTEDVALUE('QoS_Parameter(Voice)'[Index]),
    // ให้ success call setup >= 90%
    4, IF(DIVIDE(VOICE_COMPLETE + VOICE_DROP, VOICE_SAMPLE) >= 0.9 &&
DIVIDE(VOICE_COMPLETE + VOICE_DROP, VOICE_SAMPLE) <= 1.0, "#C1E1C1",
   IF(DIVIDE(VOICE_COMPLETE + VOICE_DROP, VOICE_SAMPLE) < 0.9,
"#FFB6B6", BLANK())),
   // drop call rate <= 2%
   7, IF(DIVIDE(VOICE_DROP, VOICE_SAMPLE - VOICE_BLOCK) >= 0.9 &&
DIVIDE(VOICE_DROP, VOICE_SAMPLE - VOICE_BLOCK) <= 1.0, "#FFB6B6",
   IF(DIVIDE(VOICE_DROP, VOICE_SAMPLE - VOICE_BLOCK) < 0.9, "#C1E1C1",
BLANK())),
   // successful call ratio >= 90%
   9, IF(DIVIDE(VOICE_COMPLETE, VOICE_SAMPLE) >= 0.9 &&
DIVIDE(VOICE_COMPLETE, VOICE_SAMPLE) <= 1.0, "#C1E1C1",
   IF(DIVIDE(VOICE_COMPLETE, VOICE_SAMPLE) < 0.9, "#FFB6B6", BLANK())),
   // ให้ bad call rate <= 2%
   11, IF(DIVIDE(VOICE_BAD, VOICE_SAMPLE) >= 0.9 && DIVIDE(VOICE_BAD,
VOICE_SAMPLE) <= 1.0, "#FFB6B6",
   IF(DIVIDE(VOICE_BAD, VOICE_SAMPLE) < 0.9, "#C1E1C1", BLANK()))
)

CF_TRUE_voc =
VAR VOICE_SAMPLE = CALCULATE(COUNTROWS('Database'),
'Database'[TYPE] = "Voice Successful", 'Database'[operator1] = "TRUE-H")
```

```
VAR VOICE_COMPLETE = CALCULATE(COUNTROWS('Database'), 'Database'[TYPE] = "Voice Successful", 'Database'[operator1] = "TRUE-H", 'Database'[TP] = "Complete")
VAR VOICE_BLOCK = CALCULATE(COUNTROWS('Database'), 'Database'[TYPE] = "Voice Successful", 'Database'[operator1] = "TRUE-H", 'Database'[TP] = "Block")
VAR VOICE_DROP = CALCULATE(COUNTROWS('Database'), 'Database'[TYPE] = "Voice Successful", 'Database'[operator1] = "TRUE-H", 'Database'[TP] = "Drop")
VAR VOICE_BAD = CALCULATE(COUNTROWS('Database'), 'Database'[TYPE] = "Voice Successful", 'Database'[operator1] = "TRUE-H", 'Database'[TP] = "Bad")
RETURN
SWITCH(
   SELECTEDVALUE('QoS_Parameter(Voice)'[Index]),
   // ให้ success call setup >= 90%
   4, IF(DIVIDE(VOICE_COMPLETE + VOICE_DROP, VOICE_SAMPLE) >= 0.9 && DIVIDE(VOICE_COMPLETE + VOICE_DROP, VOICE_SAMPLE) <= 1.0, "#C1E1C1",
  IF(DIVIDE(VOICE_COMPLETE + VOICE_DROP, VOICE_SAMPLE) < 0.9, "#FFB6B6", BLANK())),
  // drop call rate <= 2%
  7, IF(DIVIDE(VOICE_DROP, VOICE_SAMPLE - VOICE_BLOCK) >= 0.9 && DIVIDE(VOICE_DROP, VOICE_SAMPLE - VOICE_BLOCK) <= 1.0, "#FFB6B6",
  IF(DIVIDE(VOICE_DROP, VOICE_SAMPLE - VOICE_BLOCK) < 0.9, "#C1E1C1", BLANK())),
  // successful call ratio >= 90%
  9, IF(DIVIDE(VOICE_COMPLETE, VOICE_SAMPLE) >= 0.9 && DIVIDE(VOICE_COMPLETE, VOICE_SAMPLE) <= 1.0, "#C1E1C1",
  IF(DIVIDE(VOICE_COMPLETE, VOICE_SAMPLE) < 0.9, "#FFB6B6", BLANK())),
  // ให้ bad call rate <= 2%
  11, IF(DIVIDE(VOICE_BAD, VOICE_SAMPLE) >= 0.9 && DIVIDE(VOICE_BAD, VOICE_SAMPLE) <= 1.0, "#FFB6B6",
  IF(DIVIDE(VOICE_BAD, VOICE_SAMPLE) < 0.9, "#C1E1C1", BLANK()))
)
```