GLOBAL FINANCIAL INTELLIGENCE

DATA FLOW STRUCTURE 'S DOCUMENT FOR CRAWLER TEAM

TABLE OF CONTENTS

INTRODUCTION

Introductory document on data structure, data flow, crawler team project implementation process.

Data tables:

To know which data tables are stored in which data tables, which primary key field data, each data relation between tables, process which table data first and which later.

Functional diagram and process flow

Indicates the order of steps to be performed from top to bottom and from left to right.

Shows an overview of the processes that need to be processed on the data, the checking rules to be applied.

As a basis for building data processing functions and algorithms.

Function Interpretation and Conditional Query

Description of features, data checking rules to be applied, conditional queries for data checking...

The python programming language is used in this document to interpret functions, conditions...

Hanoi Aug 2023

Compiled by Crawler team
T/M: Tran Quang Tung
Position: Crawler & Data service support Manager
Email: tungtq@wvb.com
Tel: (+84) 931 732 381

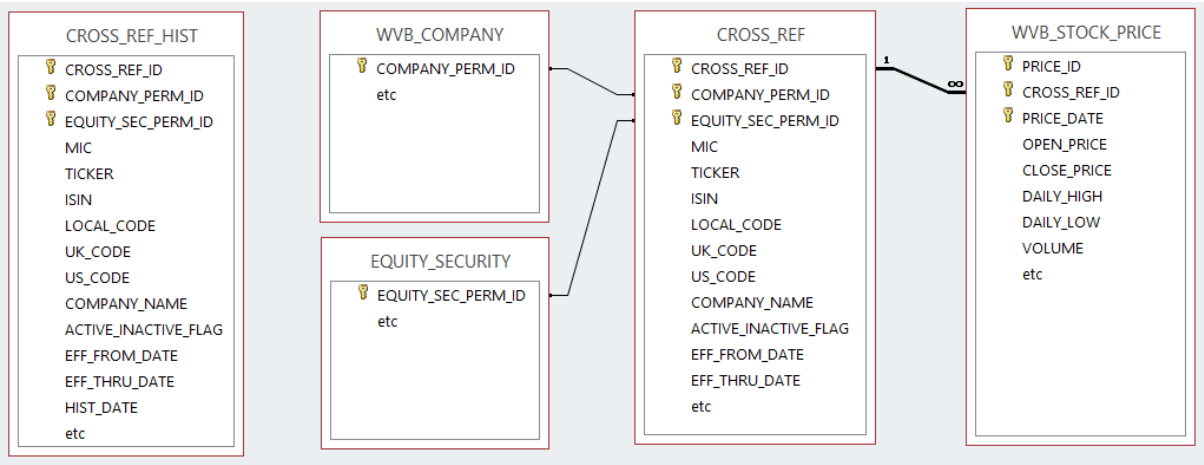
1. PRICE STOCK

Get information about stock prices: open price, high price, low price, close price, volume ... companies listed on stock exchange websites.

Data is fetched daily after session end time and historical data inserted into pre-designed tables in database.

1.1. Data tables:

Note: Diagram just list the fields is KEY and required. Main tables is CROSS_REF, WVB_STOCK_PRICE



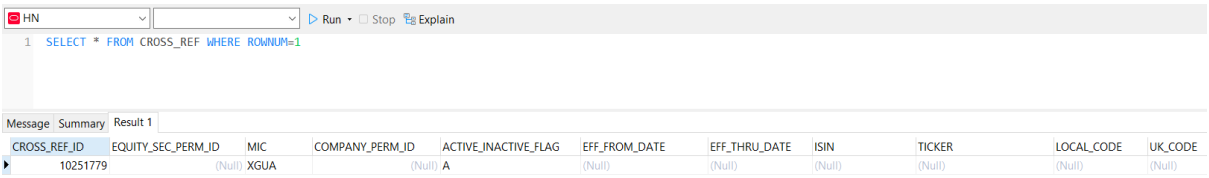
KEYS & SEQUENCE:

- CROSS_REF: CROSS_REF_SEQ => SQL: SELECT CROSS_REF_SEQ.NEXTVAL FROM DUAL

- WVB_STOCK_PRICE: PRICE_ID_SEQ => SQL: SELECT PRICE_ID_SEQ.NEXTVAL FROM DUAL

Also can view all fields of the table base on sql:

SELECT * FROM <TABLE_NAME> WHERE ROWNUM=1

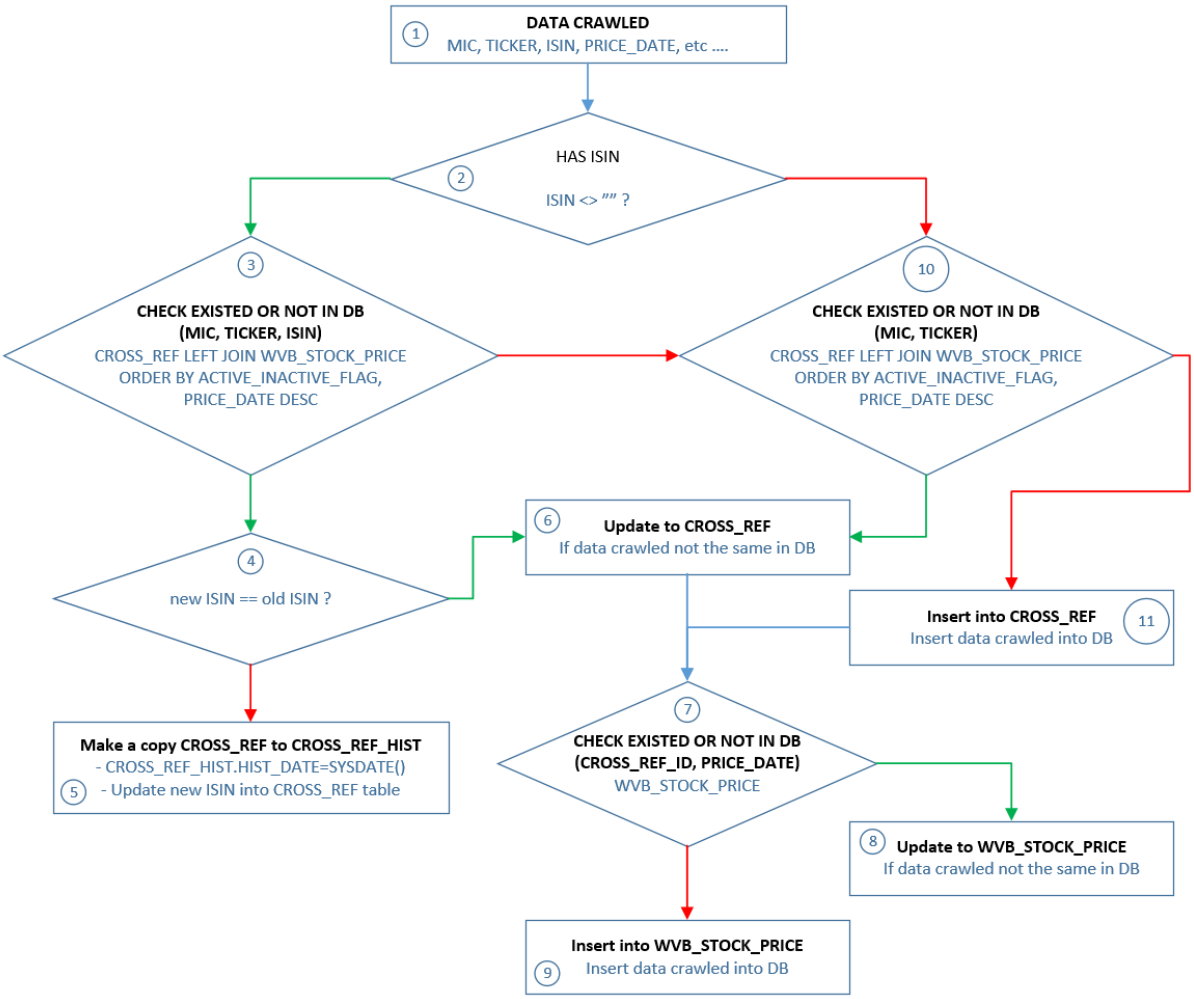


Link to database by searching COMPANY_PERM_ID (*) in table WVB_COMPANY based on MIC,TICKER,ISIN (if available) and EQUITY_SEC_PERM_ID in table EQUITY_SECURITY based on COMPANY_PERM_ID found (*) fill in CROSS_REF table if found , leave blank if not found.

1.2. Functional diagram and process flow

Components shown in graphic charts:

- **Rectangle**: process
- **Diamond**: Check condition
- **Nornal line**: Go to the next process
- **Green line**: Go to the next process if check condition result is TRUE
- **Red line**: Go to the next process if check condition result is FALSE



1.3. Function Interpretation and Conditional Query

Explanations for the functions are in numerical order circled in the diagram above (1.2).

(1). The crawled data is stored in an array named **item** for convenient description for further functions.

(2). Check whether crawled data has ISIN or not to choose the processing plan:

```
if item['ISIN'] != "":
    print("Go to step: (3)")
else:
    print("Go to step: (10)")
```

(3). Use SELECT SQL to check existed data or not in the database. Order by ACTIVE_INACTIVE_FLAG accending to get A - ACTIVE the first and I - INACTIVE is later; Order by PRICE_DATE descending to get lastest data has in the database.

SQL:

```
SELECT cr.CROSS_REF_ID, MAX(wp.Price_date) FROM cross_ref cr LEFT JOIN
wvb_stock_price wp ON cr.cross_ref_id = wp.cross_ref_id WHERE cr.MIC=
item['MIC'] AND cr.TICKER = item['TICKER'] AND cr.ISIN = item['ISIN'] GROUP BY
cr.CROSS_REF_ID ORDER BY cr.ACTIVE_INACTIVE_FLAG
```

```
# function to get data from SQL to item

def get_data_from_sql(SQL,conn):
    cur = conn.cursor()
    cur.execute(SQL)
    columns = [col[0] for col in cur.description]
    DATA=[dict(zip(columns, row)) for row in cur.fetchall()]
    cur.close()
    return DATA

# function to check existed or not

def check_existed(SQL,conn):
    cur = conn.cursor()
    cur.execute(SQL)
    rcs = cur.fetchone()

    if rcs:
        return rcs

    else:
        return None

# Get CROSS_REF_ID base on check_existed function

CROSS_REF_ID= check_existed (SQL,conn)[0] (**)

if CROSS_REF_ID:
    # Get all data from CROSS_REF to data item
    SQL="SELECT * FROM CROSS_REF WHERE CROSS_REF_ID="+str(CROSS_REF_ID)
    DATA_CROSS_REF= get_data_from_sql (SQL,conn)
    print("Go to step: (4)")
else:
    print("Go to step: (10)")
```

(4). Check ISIN crawled and ISIN in database, make copy from CROSS_REF to WORLDVEST_DEV.CROSS_REF_HIST if different

```
if item['ISIN']==DATA_CROSS_REF['ISIN']:
    print("Go to step: (6)")
else:
    print("Go to step: (5)")
```

(5). Make a copy from CROSS_REF to WORLDVEST_DEV.CROSS_REF_HIST table

```
INSERT INTO WORLDVEST_DEV.CROSS_REF_HIST SELECT * FROM CROSS_REF WHERE
CROSS_REF_ID=CROSS_REF_ID (**)
```

Update HIST_DATE to WORLDVEST_DEV.CROSS_REF_HIST table

```
UPDATE WORLDVEST_DEV.CROSS_REF_HIST SET HIST_DATE=SYSDATE WHERE
CROSS_REF_ID=CROSS_REF_ID (**)
```

(6). Update to CROSS_REF table if data crawled is different then data in the database.

```
# check to get data different between database and data crawled

def check_dif_data(DATA,item):
    DEF={}
    for k,v in DATA.items():
        if k in item and v and item[k]!=v:
            DEF[k]=v
    return DEF

CROSS_REF_DEF= check_dif_data(DATA_CROSS_REF,item)
if len(CROSS_REF_DEF)>0:
    # Update data from CROSS_REF_DEF to CROSS_REF
```

(7). Check existed or not price data in the database

```
SQL="SELECT CROSS_REF_ID FROM WVB_STOCK_PRICE WHERE
CROSS_REF_ID="+str(CROSS_REF_ID)+" AND PRICE_DATE="+item['PRICE_DATE']
PRICE_CHECK= check_existed(SQL,conn)
if PRICE_CHECK:
    print("Go to step: (8)")
else:
    print("Go to step: (9)")
```

(8). Update data into WVB_STOCK_PRICE table if has different data between crawled and data in the database.

```
SQL="SELECT * FROM WVB_STOCK_PRICE WHERE CROSS_REF_ID="+str(CROSS_REF_ID)+"
AND PRICE_DATE="+item['PRICE_DATE']
DATA_PRICE=get_data_from_sql(SQL,conn)

PRICE_DEF= check_dif_data(DATA_PRICE,item)
if len(PRICE_DEF)>0:
    # Update data from PRICE_DEF to WVB_STOCK_PRICE
```

(9). Not existed data in the WVB_STOCK_PRICE, insert data crawled into table.

(10). Continue follow (2) or (3).

```
SQL: SELECT cr.CROSS_REF_ID, MAX(wp.Price_date) FROM cross_ref cr LEFT JOIN
wvb_stock_price wp ON cr.cross_ref_id = wp.cross_ref_id WHERE cr.MIC=
item['MIC'] AND cr.TICKER = item['TICKER'] GROUP BY cr.CROSS_REF_ID ORDER BY
cr.ACTIVE_INACTIVE_FLAG

CROSS_REF_ID= check_existed (SQL,conn)[0]

if CROSS_REF_ID:
    print("Go to step: (6)")
else:
    print("Go to step: (11)")
```

(11). Not existed data in the CROSS_REF.

```
# function to get COMPANY_PERM_ID and EQUITY_SEC_PERM_ID base on ISIN
NOTE: MIC="" => MIC="~"; TICKER="" => TICKER="~"
https://dashboard.wvb.com/class/get_company_info.php?ISIN=x&MIC=x&TICKER=x
def get_more_data(MIC,TICKER,ISIN,conn):
    if MIC=="" or MIC is None:
        MIC="~"
    if TICKER=="" or TICKER is None:
        TICKER="~"
    item={}
    cur = conn.cursor()
sql="SELECT (SELECT OFFICE FROM WVBADMIN.HEADERSOFFICE o WHERE
o.COMPANY_PERM_ID=wc.company_perm_id AND ROW_NO = 1) OFFICE,
WC.COMPANY_PERM_ID, ES.EQUITY_SEC_PERM_ID, WC.WVB_NUMBER,
WC.PRIMARY_SHORT_COMPANY_NAME, WC.PRIMARY_LONG_COMPANY_NAME,
ES.ISIN_FROM_REPORT,WC.RECORD_TYPE,WC.COMPANY_TYPE,WC.IND_GROUPING_TYPE,WC.UPD
ATING_PRIORITY_CODE,ES.PRIMARY_STOCK_EXCHG_ALPHA_CODE,
ES.PRIMARY_TICKER,ES.FIRST_SEACODE_FOR_PRICES,ES.FIRST_TICKER,ES.SECOND_SEACOD
E_FOR_PRICES,ES.SECOND_TICKER,
ES.THIRD_SEACODE_FOR_PRICES,ES.THIRD_TICKER,ES.FOURTH_SEACODE_FOR_PRICES,ES.FO
URTH_TICKER, WC.WVB_STAT_CODE_INTERNAL,  ES.SEC_CLASSIF_CODE,
decode(WC.WVB_STAT_CODE_INTERNAL, '90', 1, '11', 2, '15', 3, '91', 4, 100)
code_order, decode( ES.SEC_CLASSIF_CODE, 'ORD', 1, 100) type_order,
NVL2(t.wvb_number, 'YES','NO') top5000, NVL2(e.wvb_number, 'YES','NO') ecb
FROM WVB_COMPANY WC  JOIN EQUITY_SECURITY ES ON WC.COMPANY_PERM_ID =
ES.COMPANY_PERM_ID  LEFT JOIN TUNGTQ.TOP5000 t ON t.WVB_NUMBER = wc.WVB_NUMBER
AND t.EFF_END_DATE >= SYSDATE LEFT JOIN TUNGTQ.ECB e ON e.WVB_NUMBER =
wc.WVB_NUMBER  AND e.EFF_END_DATE >= SYSDATE WHERE
TRIM(UPPER(ES.ISIN_FROM_REPORT)) = TRIM(UPPER('"+ISIN+"'))  OR
(TRIM(UPPER(ES.PRIMARY_STOCK_EXCHG_ALPHA_CODE))=TRIM(UPPER('"+MIC+"')) AND
TRIM(UPPER(ES.PRIMARY_TICKER))=TRIM(UPPER('"+TICKER+"'))) OR
(TRIM(UPPER(ES.FIRST_SEACODE_FOR_PRICES))=TRIM(UPPER('"+MIC+"')) AND
TRIM(UPPER(ES.FIRST_TICKER))=TRIM(UPPER('"+TICKER+"'))) OR
(TRIM(UPPER(ES.SECOND_SEACODE_FOR_PRICES))=TRIM(UPPER('"+MIC+"')) AND
TRIM(UPPER(ES.SECOND_TICKER))=TRIM(UPPER('"+TICKER+"'))) OR
(TRIM(UPPER(ES.THIRD_SEACODE_FOR_PRICES))=TRIM(UPPER('"+MIC+"')) AND
TRIM(UPPER(ES.THIRD_TICKER))=TRIM(UPPER('"+TICKER+"'))) OR
(TRIM(UPPER(ES.FOURTH_SEACODE_FOR_PRICES))=TRIM(UPPER('"+MIC+"')) AND
TRIM(UPPER(ES.FOURTH_TICKER))=TRIM(UPPER('"+TICKER+"'))) ORDER BY code_order,
type_order,  ES.SEC_CLASSIF_CODE desc"

    cur.execute(sql)
    row = cur.fetchone()
    if row:
        item['COMPANY_PERM_ID']=str(row[0])
        item['EQUITY_SEC_PERM_ID']=str(row[1])
    cur.close()
    return item
```

```
if item['ISIN'] or (item["MIC"] and item["TICKER"]):
    COM=get_more_data(MIC,TICKER,ISIN,conn)
    item['COMPANY_PERM_ID']=COM['COMPANY_PERM_ID']
    item['EQUITY_SEC_PERM_ID']=COM['EQUITY_SEC_PERM_ID']
Insert data crawled into CROSS_REF table. Go back step (7).
```

2. PRICE INDEX

Get information about indies of stock: open value, high value, low value, close value, volume ... listed on stock exchange websites.

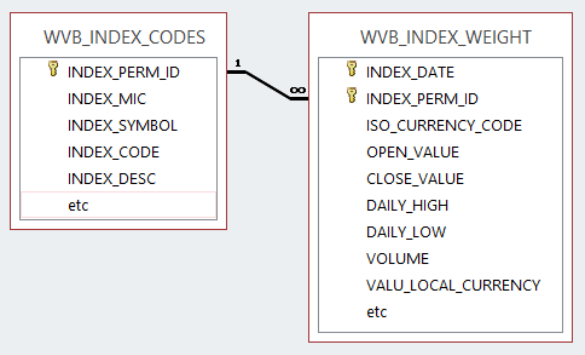
Data is fetched daily after session end time and historical data inserted into pre-designed tables in database.

2.1. Data tables:

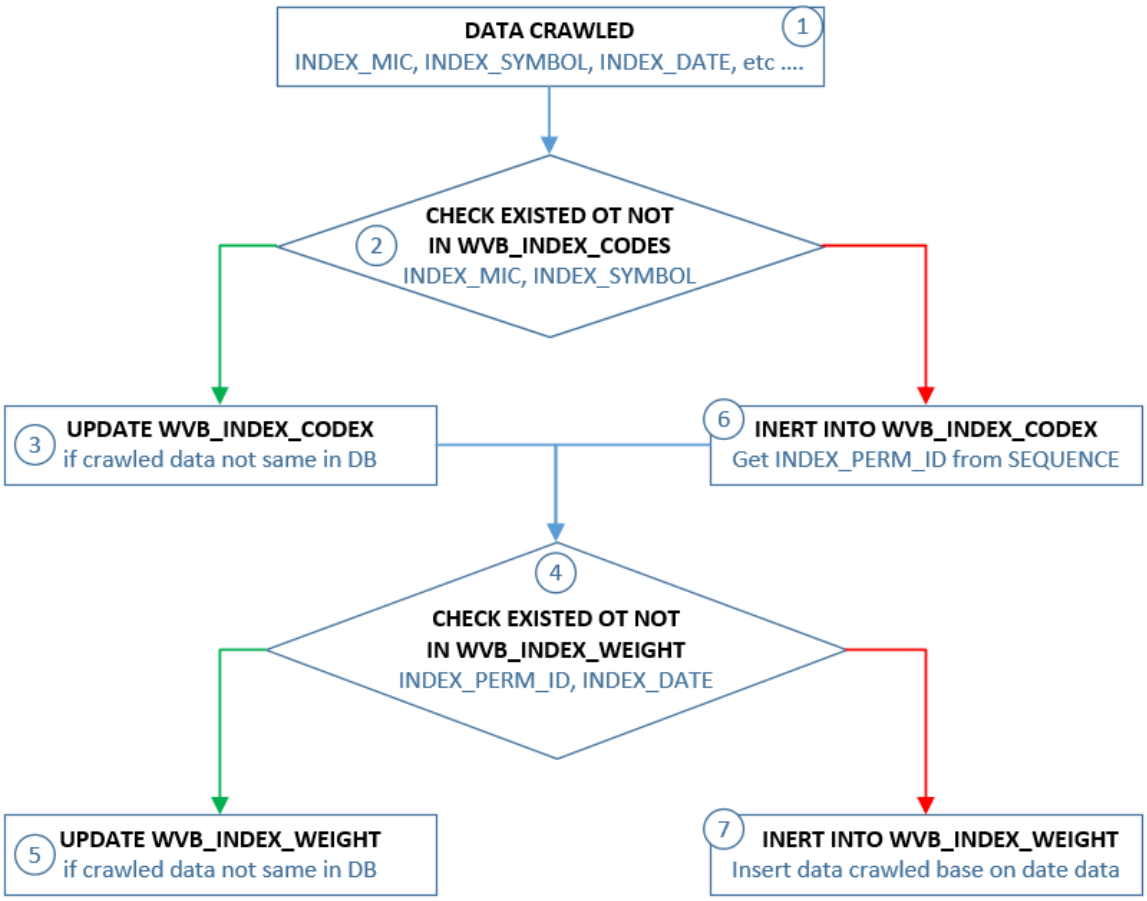
<u>Note:</u> Diagram just list the fields is KEY and required. Main tables is WVB_INDEX_CODE, WVB_INDEX_WEIGHT.

KEYS & SEQUENCE:

- WVB_INDEX_CODES: INDEX_PERM_ID



2.2. Functional diagram and process flow



2.3. Function Interpretation and Conditional Query

(1). The crawled data is stored in an array named **item** for convenient description for further functions.

(2). Check data crawled existed or not in the table WVB_INDEX_CODES base on INDEX_MIC, INDEX_SYMBOL.

```
SELECT ID.INDEX_PERM_ID,MAX(IW.INDEX_DATE) FROM WVB_INDEX_CODES ID LEFT JOIN
WVB_INDEX_WEIGHT IW ON ID.INDEX_PERM_ID=IW.INDEX_PERM_ID WHERE
ID.INDEX_MIC=item['INDEX_MIC'] AND ID.INDEX_SYMBOL=item['INDEX_SYMBOL'] GROUP
BY ID.INDEX_PERM_ID

# Get INDEX_PERM_ID base on check_existed function
INDEX_PERM_ID= check_existed (SQL,conn)[0] (**)

if INDEX_PERM_ID:
    # Get all data from CROSS_REF to data item
    SQL="SELECT * FROM WVB_INDEX_CODES WHERE INDEX_PERM_ID="+str(INDEX_PERM_ID)
    DATA_INDEX_CODES= get_data_from_sql (SQL,conn)
    print("Go to step: (3)")
else:
    print("Go to step: (6)")
```

(3). Update WVB_INDEX_CODES if data crawled is different with DB

```
INDEX_CODE_DEF= check_dif_data(DATA_CROSS_REF,item)
if len(INDEX_CODE_DEF)>0:
    # Update data from INDEX_CODE_DEF to WVB_INDEX_CODES
```

(4). Check data crawled and data in WVB_INDEX_WEIGHT.

```
SQL="SELECT * FROM WVB_INDEX_WEIGHT WHERE INDEX_PERM_ID="+str(INDEX_PERM_ID)+"
AND INDEX_DATE="+item['INDEX_DATE']
DATA_WEIGHT=get_data_from_sql(SQL,conn)
WEIGHT_DEF= check_dif_data(DATA_WEIGHT,item)
(5),(7)
if len(WEIGHT_DEF)>0:
    # Update data from WEIGHT_DEF to WVB_INDEX_WEIGHT (5)

else:
    # Update data from WEIGHT_DEF to WVB_INDEX_WEIGHT (7)
```

(6). Insert new data crawled in to WVB_INDEX_CODES.

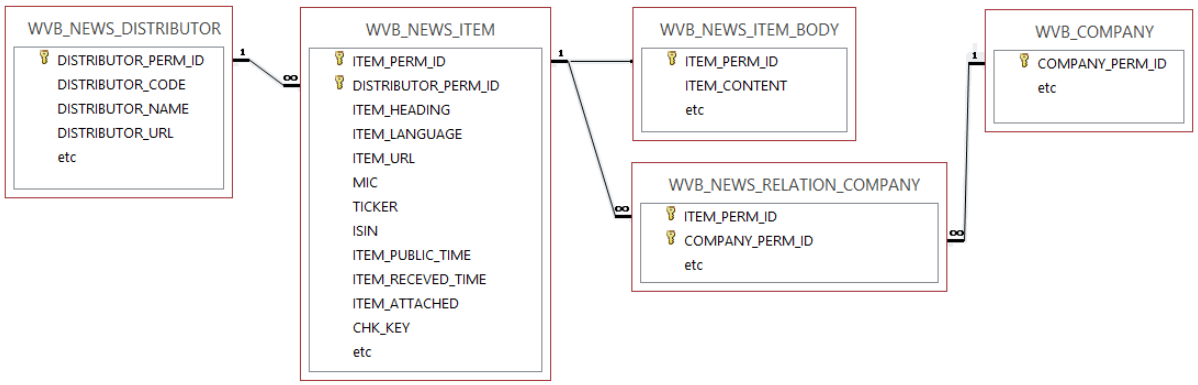
3. NEWSFEEDS

Crawl news from website and export to json data base on those array keys required:

- WEBSITE_NAME: The name of the website
- ITEM_HEADING: The title of the article
- ITEM_LANGUAGE: The language of the article as ISO CODE (“EN”,”VI”,”FR” ….)
- ITEM_URL: The url of the article
- MIC, TICKER, ISIN of the company if has
- ITEM_PUBLIC_TIME: The data time public of article

- ITEM_RECEIVED_TIME: The data time crawl
- ITEM_ATTACHED: The list url of attach document if has (As: [“url1”,”url2”,”url…”])
- ITEM_CONTENT: The body of the article

3.1. Data tables:

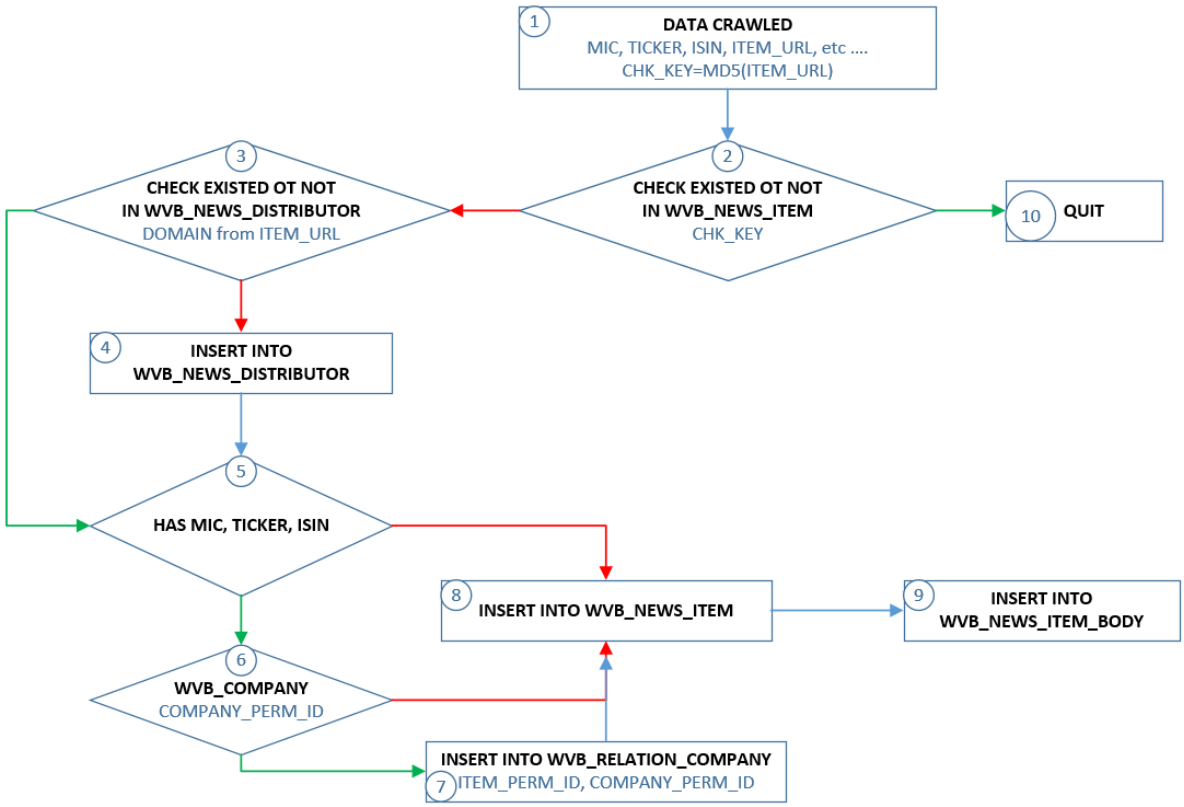


KEYS & SEQUENCE:

- ITEM_PERM_ID: WVB_NEWS_SEQ

- ATTACHMENT_PERM_ID: NEWS_ATTACHMENT_SEQ

3.2. Functional diagram and process flow



3.3. Function Interpretation and Conditional Query

(1). The crawled data is stored in an array named **item** for convenient description for further functions.

```
import hashlib

def key_MD5(txt):
    txt=(txt.upper()).strip()
    results=hashlib.md5(txt.encode('utf-8')).hexdigest()
    return results

CHK_KEY=key_MD5(item['ITEM_URL'])
```

(2). Check data crawled existed or not in the table WVB_NEWS_ITEM base on CHK_KEY.

```
SQL: SELECT ITEM_PERM_ID FROM WVB_NEWS_ITEM WHERE CHK_KEY='xxx'
ITEM_PERM_ID = check_existed (SQL,conn)[0]

if ITEM_PERM_ID:
    print("Existed -> QUIT")
else:
    print("Go to step: (3)")
```

(3) Check existed or not in WVB_NEWS_DISTRIBUTOR base on domain name

```
DOMAIN=str(item['ITEM_URL']).split('/')[2]
SQL: SELECT DISTRIBUTOR_PERM_ID FROM WVB_NEWS_DISTRIBUTOR WHERE DISTRIBUTOR_URL
LIKE '%www.szse.cn%'
DISTRIBUTOR_PERM_ID = check_existed (SQL,conn)[0]
if DISTRIBUTOR_PERM_ID:
    print("Insert into WVB_NEWS_ITEM: (5)")
else:
    print("Insert into WVB_NEWS_DISTRIBUTOR: (4)")
```

(4). Insert data into WVB_NEWS_DISTRIBUTOR

(5). If has MIC, TICKER or ISIN data crawled

```
if item['MIC']!='' and (item['TICKER']!='' or item['ISIN']!=''):
    print('Go to step: (8)')
else:
    print('QUIT')
```

(6). Get company information base one MIC, TICKER, ISIN

```
if item['ISIN'] or (item["MIC"] and item["TICKER"]):
    COM=get_more_data(MIC,TICKER,ISIN,conn)
    item['COMPANY_PERM_ID']=COM['COMPANY_PERM_ID']
    item['WVB_NUMBER']=COM['WVB_NUMBER']
```

(7). Insert data into WVB_RELATION_COMPANY

(8). Insert data into WVB_NEWS_ITEM

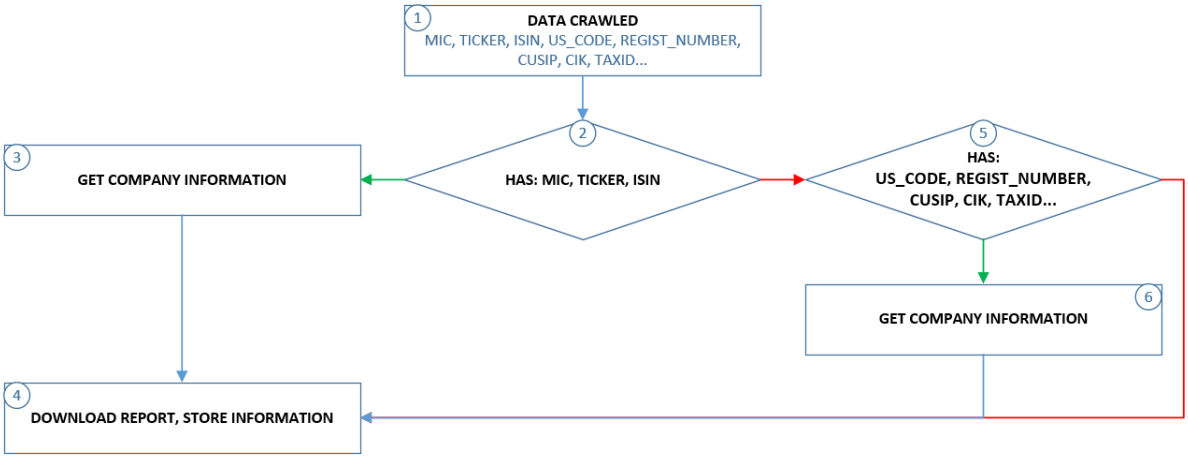
(9). Insert data into WVB_NEWS_ITEM_BODY

(10). Quit

4. DOWNLOAD REPORT

The crawl data tries to get by Company for information including (if possible): MIC, TICKER, ISIN or US_CODE, REGIST_NUMBER, CUSIP, CIK, TAXID...(X)

4.1. Functional diagram and process flow



4.2. Function Interpretation and Conditional Query

(1). The crawled data is stored in an array named **item** for convenient description for further functions.

(2). Base on data crawled to map and get company information from database.

```
if item['ISIN'] or (item["MIC"] and item["TICKER"]):
    print('Go to step: (3)')
else:
    print('Go to step: (5)')
```

(3). Get company information base on MIC, TICKER, ISIN.

```
https://dashboard.wvb.com/class/get_company_info.php?ISIN=x&MIC=x&TICKER=x
or    COMPANY_INFO=get_more_data(MIC,TICKER,ISIN,conn)
=> Go to step 4.
```

(5). Get company information base on US_CODE, REGIST_NUMBER, CUSIP, CIK, TAXID...

```
SQL="SELECT COMPANY_PERM_ID FROM COMPANY_ALIAS_NAME WHERE ALIAS_NAME_TYPE='X'
AND COMPANY_NAME='VALUE'"

COMPANY_PERM_ID = check_existed (SQL,conn)[0]

if COMPANY_PERM_ID:
    print("Go to step: (6)")
else:
    print("Go to step: (4)")
```

```
(6). Get company information base on COMPANY_PERM_ID

https://dashboard.wvb.com/class/get_company_info.php?ID=x

=> Go to step 4.
```

(4). Download report, store data:

FILE NAME STRUCTURE (suggestion):

PRIORITY~GROUPPING_TYPE~WVB_NUMBER~ECB~TOP5K~<REPORT_DATE>~REPORT_NAME_..

Example: P1~BANK~USA000037082~NO~YES~2023-09-05~Annual_report_xxx.pdf

Company information: https://dashboard.wvb.com/class/get_company_info.php?ID=2551212

5. PRIVATE COMPANY

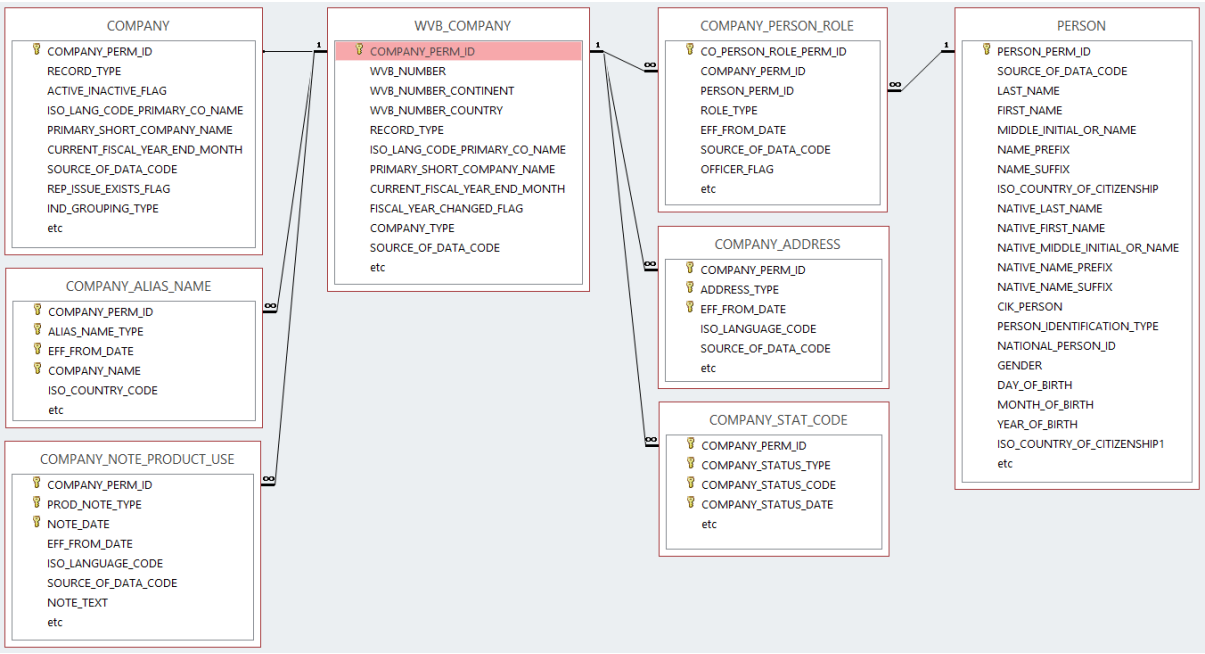
Try to get as much information as possible from the source website, including the following groups of information (english and local language together):

- Company Information: Company name, company identity: MIC, TICKER, ISIN, US_CODE, REGIST_NUMBER, CUSIP, CIK, TAXID..., company address, registered business lines …
- Persons: Person name, Address, Position, ID card number, Passpost number, Driver's license number …

Note: An individual working at this company may also hold another position at another company, or may have held a position at a previous company. Needs to be defined to avoid duplication.

5.1. Data tables:

Note: Diagram just list the fields is KEY and required. Main tables is WVB_COMPANY, PERSON

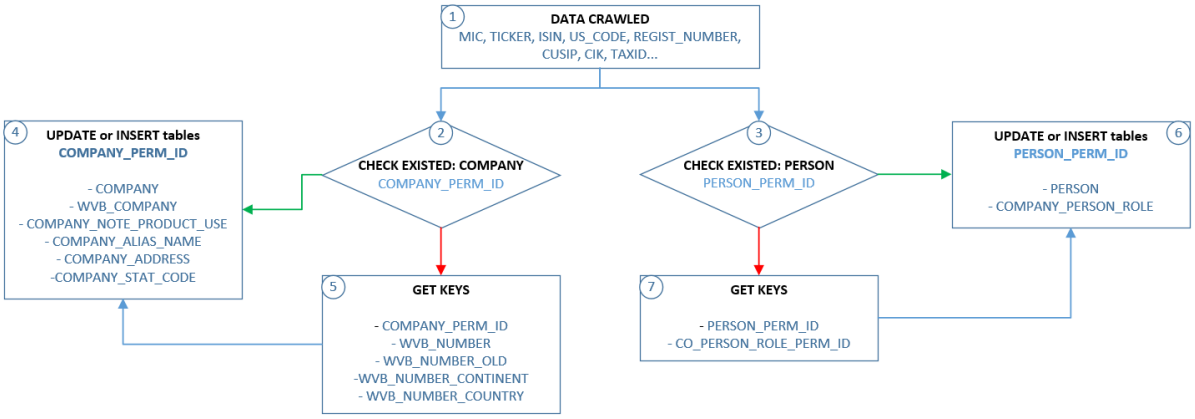


KEYS & SEQUENCE:

- COMPANY_PERM_ID: COMPANY_PERM_ID - PERSON_PERM_ID: PERSON_PERM_ID_SEQ

- CO_PERSON_ROLE_PERM_ID: CO_PERSON_ROLE_PERM_ID_SEQ

5.2. Functional diagram and process flow



5.3. Function Interpretation and Conditional Query

(1). The crawled data is stored in an array named **item** for convenient description for further functions.

```
def Get_ID(FIELD,conn):
    cur = conn.cursor()
    cur.execute("SELECT "+FIELD+".NEXTVAL FROM DUAL")
    ID=cur.fetchone()[0]
    cur.close()
    return ID
def Get_WVB_NUMBER(country_code,conn):
    cur = conn.cursor()
    cur.execute("SELECT WVB_NUMBER_SEQ + 1,WVB_COUNTRY_CODE FROM COUNTRY_CODES
WHERE ISO_COUNTRY_CODE='"+country_code+"'")
    rcs=cur.fetchone()
    RESULT={'WVB_NUMBER':rcs[0],'COUNTRY_CODE':rcs[1]}
    cur.execute("UPDATE COUNTRY_CODES SET WVB_NUMBER_SEQ  = WVB_NUMBER_SEQ +1
WHERE ISO_COUNTRY_CODE ='"+country_code+"'")
    conn.commit()
    cur.close()
    return RESULT

# Get KEYS
COMPANY_PERM_ID=Get_ID('COMPANY_PERM_ID', self.conn)
WVB_NUM=Get_WVB_NUMBER('KHM', self.conn)
WVB_NUMBER_OLD=str(WVB_NUM['WVB_NUMBER'])+WVB_NUM['COUNTRY_CODE']
WVB_NUMBER='KHM'+("%09d" % (WVB_NUM['WVB_NUMBER']))
WVB_NUMBER_CONTINENT=WVB_NUM['COUNTRY_CODE'][0]
WVB_NUMBER_COUNTRY=WVB_NUM['COUNTRY_CODE'][1]
```

(2). Check existed or not Company information base on steps: 2,3,5,6 of 4.2;

ALIAS name and type base on COMPANY_ALIAS_NAME and ALIAS_NAME_TYPES tables

Sample base on iamge and table below:

PROD_TUNGTQ | WORLDVEST_DEV | Run | Stop | Explain

```
1  SELECT * FROM COMPANY_ALIAS_NAME WHERE COMPANY_PERM_ID=10025926
```

Message | Summary | Result 1

COMPANY_PERM_ID	ALIAS_NAME_TYPE	COMPANY_NAME
10025926	CIK	0001138723
10025926	STOCKEX	ARAY
10025926	LEGAL	ACCURAY INCORPORATED
10025926	PRODUCT	ACCURAY
10025926	TAXID	208370041
10025926	STOCKEX	XEJ
10025926	CUSIP	004397105
10025926	REGIST	3358338

ALIAS_NAME_TYPE	ALIAS_NAME_DESC
TAXID	TAX PAYER ID
NATIVE	LEGAL COMPANY NAME IN LOCAL LANGUAGE (NON ROMAN CHARACTERS)
PRVNATIV	PREVIOUS NATIVE NAME OF COMPANY
FDICID	FDIC ID
COMMON	COMMON USAGE NAME
JCQT18	JCQT NAME (18 CHARATER NAME)
ALIAS	ALIAS NAME
LEGAL	LEGAL NAME
ENGLISH	ENGLISH LANGUAGE NAME
STOCKEX	LOCAL STOCK EXCHANGE TICKER SYMBOL
LABEL	NAME USED ON ADDRESS LABEL
PRODUCT	NAME USED ON PRODUCT - PRIMARY COMPANY SHORT NAME
PRVLEGAL	PREVIOUS LEGAL NAME
REUTERS	REUTERS
TELEKURS	TELEKURS
PRVENG	PREVIOUS ENGLISH NAME OF COMPANY
CUSIP	CUSIP
REGIST	COMPANY LOCAL REGISTRATION NUMBER
CIK	CENTRAL INDEX KEY _US_SEC
STOCKLG	NAME USED ON STOCK EXCHANGE
SICOVAM	SICOVAM PARIS
MNEMO	MNEMO PARIS
CIB	CENTRAL INDEX KEY - CANADA
BVD	BVD COMPANY ID
GST_VAT	GOODS AND SERVICES TAX ID / VALUE-ADDED TAX ID
CIN	CORPORATE IDENTITY NUMBER
REGISTMA	MALAYSIAN NEW COMPANY LOCAL REGISTRATION NUMBER (EFF 2019)
CU	CREDIT UNION CHARTER NUMBER
SWIFT	SOCIETY FOR WORLDWIDE INTERBANK FINANCIAL TELECOMMUNICATION (SWIFT) / BANK IDENTIFIER CODES (BIC)

(3) Check existed PERSON base on Person Name, Persion ID, Person Brith …

```
SQL="SELECT * FROM PERSON WHERE <<FIELDS>> = <<VALUES>>"

PERSON_PERM_ID= check_existed (SQL,conn)[0]

(5), (7): use function base on step (1).

(4), (6): Update or insert into tables.
```

Note: can use this lib to get Company product name base on Legal name: pip install cleanco

```
>>> from cleanco import basename
>>> business_name = "Some Big Pharma, LLC"
>>> basename(business_name)
>>> 'Some Big Pharma'
```

The document completed in September 2023 based on the current database structure and data matching measures, and will be updated if there are future changes.

Link: https://dashboard.wvb.com/data/CRAWLER_DATA_FLOW_STRUCTURE.pdf

--- THE END ---