[ ]:
```
# Author: Birsan Daniel
# Title: Company Entity Deduplication

import re
import pandas as pd
import numpy as np

import pycountry
import tldextract
import spacy
from transformers import pipeline

import dask.dataframe as dd

# 1. Imports & Model Initialization
nlp_spacy = spacy.load("en_core_web_sm")


extractor = pipeline(
    "token-classification",
    model="dslim/bert-base-NER",
    aggregation_strategy="simple",
    device=0  # set device=-1 if no GPU
)


stopwords_suffixes = [
    "inc", "inc.", "co", "corp", "corporation", "ltd", "llc", "srl", "gmbh",␣
 ↪"ltda", "limited"
]

# 2. Load Data
try:
    df_dask = dd.read_parquet(
        "veridion_entity_resolution_challenge.snappy.parquet").persist()
    df_raw = df_dask.compute()
    print("Using Dask to load large dataset.")
```

```
except (FileNotFoundError, ValueError):
    print("Could not load with Dask, falling back to pandas.")
    try:
        df_raw = pd.read_parquet(
            "veridion_entity_resolution_challenge.snappy.parquet")
    except FileNotFoundError:
        print("veridion_entity_resolution_challenge.snappys.parquet not found;␣
 ↪creating empty DataFrame.")
        df_raw = pd.DataFrame()

print("Raw DataFrame Loaded:")
print(f"Shape: {df_raw.shape}")

# 3. Helper Functions


def remove_company_suffixes(name: str, suffixes=stopwords_suffixes) -> str:
    if name is None or not isinstance(name, str) or not name.strip():
        return None
    tokens = name.split()
    cleaned_tokens = []
    for t in tokens:
        if t.lower() in suffixes:
            continue
        cleaned_tokens.append(t)
    return " ".join(cleaned_tokens).strip()


def normalize_text(text: str) -> str:
    if text is None or not isinstance(text, str) or not text.strip():
        return None
    out = text.strip().lower()
    out = re.sub(r"\s+", " ", out)
    return out


def normalize_email(email: str) -> str:
    if email is None or not isinstance(email, str):
        return None
    e = email.strip().lower()
    return e if "@" in e else None


def normalize_phone(phone: str) -> str:
    if phone is None or not isinstance(phone, str):
        return None
    p = phone.strip()
```

```
    p = re.sub(r"[^\d\+]", "", p)
    return p


def parse_domain_tld(url: str):
    if url is None or not isinstance(url, str) or not url.strip():
        return (None, None)
    extracted = tldextract.extract(url)
    #only the name , no tld, no subdomain
    domain = extracted.domain.strip().lower()
    tld = extracted.suffix.strip().lower()
    if not domain or not tld:
        return (None, None)
    if domain.endswith("."):
        domain = domain[:-1]
    if tld.endswith("."):
        tld = tld[:-1]
    return (domain, tld)


def approximate_country_from_code(code: str) -> str:
    if code is None or not code.strip():
        return None
    up = code.strip().upper()
    cobj = pycountry.countries.get(alpha_2=up)
    if not cobj:
        cobj = pycountry.countries.get(alpha_3=up)
    return cobj.name if cobj else None


def approximate_country_from_tld(tld: str) -> str:
    if tld is None or not tld.strip():
        return None
    if len(tld) == 2:
        return approximate_country_from_code(tld)
    return None


def extract_company_name_spacy(text: str) -> str:
    if text is None or not isinstance(text, str) or not text.strip():
        return None
    doc = nlp_spacy(text)
    for ent in doc.ents:
        if ent.label_ == "ORG":
            return ent.text.strip()
    return None
```

```python
def extract_activity_transformers(text: str) -> str:
    if text is None or not isinstance(text, str) or not text.strip():
        return None
    results = extractor(text)
    found = []
    for r in results:
        if r['entity_group'] in ['MISC', 'ORG']:
            extracted_word = r['word'].strip().lower()
            found.append(extracted_word)
    if found:
        unique_found = list(set(found))
        return ", ".join(unique_found)
    return None


def get_first_n_words(text: str, n: int = 3) -> str:
    if text is None or not isinstance(text, str) or not text.strip():
        return None
    words = text.strip().split()
    return " ".join(words[:n]) if words else None


# ------------------------------------------------------------------------------
# 4. Check & Create Missing Columns
# ------------------------------------------------------------------------------
needed_cols = [
    # Company Identifiers:
    "company_name", "company_legal_names", "company_commercial_names",

    # Location Data:
    "main_country_code", "main_country", "main_region", "main_city_district",␣
 ↪"main_city", "main_postcode", "main_street", "main_street_number",␣
 ↪"main_latitude", "main_longitude", "main_address_raw_text", "locations",

    # Contact Information:
    "primary_phone", "phone_numbers", "primary_email", "emails", "other_emails",

    # Web Presence:
    "website_url", "website_domain", "website_tld", "website_language_code",␣
 ↪"facebook_url", "twitter_url", "instagram_url", "linkedin_url",␣
 ↪"ios_app_url", "android_app_url", "youtube_url", "tiktok_url",

    # Business Details:
    "company_type", "year_founded", "lnk_year_founded", "short_description",␣
 ↪"long_description", "business_tags", "business_model", "product_type",
```

```
    # Industry Codes & Categories:
    "naics_vertical", "naics_2022_primary_code", "naics_2022_primary_label",␣
 ↪"naics_2022_secondary_codes", "naics_2022_secondary_labels",␣
 ↪"main_business_category", "main_industry", "main_sector",

    # Other Data:
    "sics_codified_industry", "sic_codes", "sic_labels", "isic_v4_codes",␣
 ↪"isic_v4_labels", "nace_rev2_codes", "nace_rev2_labels", "created_at",␣
 ↪"last_updated_at", "website_number_of_pages", "generated_description",␣
 ↪"generated_business_tags", "status", "domains", "all_domains", "revenue",␣
 ↪"revenue_type", "employee_count", "employee_count_type",␣
 ↪"inbound_links_count"
]

for col in needed_cols:
    if col not in df_raw.columns:
        df_raw[col] = None


# 5. Company Name Enrichment


def best_company_name(row: dict) -> str:
    web = row.get("website_url")
    # handle pd.NA by converting to None if it's a float or pd.NA
    if isinstance(web, float) or pd.isna(web):
        web = None

    domain = None
    if web:
        d, s = parse_domain_tld(web)
        domain = d

    if domain:
        name_candidate = normalize_text(domain)
        return remove_company_suffixes(name_candidate)

    for desc_col in ["short_description", "long_description",␣
 ↪"generated_description"]:
        desc_val = row.get(desc_col)
        if isinstance(desc_val, float) or pd.isna(desc_val):
            desc_val = None
        if desc_val:
            first3 = get_first_n_words(desc_val, 3)
            first3 = normalize_text(first3)
            if first3:
                return remove_company_suffixes(first3)
```

```
    name_candidates = [
        row.get("company_name"),
        row.get("company_legal_names"),
        row.get("company_commercial_names"),
    ]
    for candidate in name_candidates:
        if isinstance(candidate, float) or pd.isna(candidate):
            candidate = None
        candidate_norm = normalize_text(candidate)
        if candidate_norm:
            return remove_company_suffixes(candidate_norm)

    # spaCy fallback
    for desc_col in ["short_description", "long_description",␣
 ↪"generated_description"]:
        desc_val = row.get(desc_col)
        if isinstance(desc_val, float) or pd.isna(desc_val):
            desc_val = None
        ner_name = extract_company_name_spacy(desc_val)
        if ner_name:
            final = normalize_text(ner_name)
            return remove_company_suffixes(final)

    return None

# 6. Location Enrichment (Offline)
# external geocoding will need more time :*(


def fill_missing_location(row: dict) -> dict:
    country = row.get("main_country")
    code = row.get("main_country_code")

    if isinstance(country, float) or pd.isna(country):
        country = None
    if isinstance(code, float) or pd.isna(code):
        code = None

    if not country and code:
        py_c = approximate_country_from_code(code)
        if py_c:
            country = py_c

    if not country:
        for ccol in ["website_url", "website_domain"]:
            val = row.get(ccol)
```

```
            if isinstance(val, float) or pd.isna(val):
                val = None
            if val:
                d, s = parse_domain_tld(val)
                guess = approximate_country_from_tld(s)
                if guess:
                    country = guess
                    break

    row["main_country"] = normalize_text(
        country) if country else row["main_country"]
    return row


def fill_city_from_postcode(row: dict) -> dict:

    return row

def unify_activity_info(row: dict) -> dict:
    code_cols = [
        "sics_codified_industry",
        "sic_codes", "sic_labels",
        "isic_v4_codes", "isic_v4_labels",
        "nace_rev2_codes", "nace_rev2_labels",
        "naics_2022_primary_code", "naics_2022_primary_label",
        "naics_2022_secondary_codes", "naics_2022_secondary_labels",
        "naics_vertical",
        "business_tags",
    ]

    activity_set = set()
    for c in code_cols:
        val = row.get(c)
        if pd.isna(val) or isinstance(val, float):
            val = None
        if val and isinstance(val, str) and val.strip():
            splitted = [x.strip() for x in val.split(",")]
            for s in splitted:
                if s:
                    activity_set.add(s.lower())

    if not activity_set:
        for desc_col in ["short_description", "long_description",␣
 ↪"generated_description"]:
            desc_val = row.get(desc_col)
            if pd.isna(desc_val) or isinstance(desc_val, float):
                desc_val = None
```

```
            if desc_val and desc_val.strip():
                hf_activity = extract_activity_transformers(desc_val)
                if hf_activity:
                    for token in hf_activity.split(","):
                        token = token.strip()
                        if token:
                            activity_set.add(token)

    if activity_set:
        row["activity_enriched"] = ", ".join(sorted(activity_set))
    else:
        row["activity_enriched"] = None

    # Safely check for main_business_category
    mbc = row.get("main_business_category")
    if pd.isna(mbc) or isinstance(mbc, float) or (not mbc):
        # only fill if we have activity_enriched
        if row["activity_enriched"]:
            row["main_business_category"] = next(iter(activity_set))

    # Safely check for main_industry
    mind = row.get("main_industry")
    if pd.isna(mind) or isinstance(mind, float) or (not mind):
        if row["activity_enriched"]:
            row["main_industry"] = next(iter(activity_set))

    return row


def normalize_social(url: str) -> str:
    if url is None or not isinstance(url, str) or not url.strip():
        return None
    s = url.strip().lower()
    s = re.sub(r"^https?://", "", s)
    return s.rstrip("/")

# 8. Parallel Execution with meta


def apply_all_transformations(df: pd.DataFrame) -> pd.DataFrame:
    df = df.copy()

    df["company_name"] = df.apply(best_company_name, axis=1)

    df = df.apply(fill_missing_location, axis=1)
    df = df.apply(fill_city_from_postcode, axis=1)
```

```
    location_cols = [
        "main_country", "main_region", "main_city_district",
        "main_city", "main_postcode", "main_street",
        "main_street_number", "main_address_raw_text"
    ]
    for col in location_cols:
        df[col] = df[col].apply(normalize_text)

    df = df.apply(unify_activity_info, axis=1)

    df["primary_email"] = df["primary_email"].apply(normalize_email)
    df["emails"] = df["emails"].apply(normalize_email)
    df["other_emails"] = df["other_emails"].apply(normalize_email)

    df["primary_phone"] = df["primary_phone"].apply(normalize_phone)
    df["phone_numbers"] = df["phone_numbers"].apply(normalize_phone)

    for sc in ["facebook_url", "twitter_url", "instagram_url", "linkedin_url",
               "ios_app_url", "android_app_url", "youtube_url", "tiktok_url"]:
        df[sc] = df[sc].apply(normalize_social)

    df["revenue"] = pd.to_numeric(df["revenue"], errors="coerce")
    df["employee_count"] = pd.to_numeric(df["employee_count"], errors="coerce")

    return df


def create_meta(df_sample: pd.DataFrame) -> pd.DataFrame:
    """
    Create a zero-row DataFrame with the same columns/dtypes as final output.
    """
    if df_sample.empty:
        meta_cols = {
            "company_name": "object",
            "activity_enriched": "object",
            "main_country": "object",
            "main_region": "object",
            "main_city": "object",
        }
        meta_df = pd.DataFrame({col: pd.Series(dtype=dt)
                                for col, dt in meta_cols.items()})
        return meta_df.iloc[:0]
    else:
        sample = df_sample.head(1).copy()
        for col in ["company_name", "activity_enriched"]:
            if col not in sample.columns:
                sample[col] = pd.Series(dtype="object")
```

```
            else:
                sample[col] = sample[col].astype("object")
        return sample.iloc[:0].copy()

# ------------------------------------------------------------------------------
# 9. Final Output
# ------------------------------------------------------------------------------


if len(df_raw) > 5000:
    ddf = dd.from_pandas(df_raw, npartitions=8)
    meta_df = create_meta(df_raw)
    ddf_transformed = ddf.map_partitions(
        apply_all_transformations, meta=meta_df)
    df_cleaned = ddf_transformed.compute()
else:
    df_cleaned = apply_all_transformations(df_raw)

print("\nCleaned & Enriched DataFrame (Sample):")
print(f"Shape: {df_cleaned.shape}")
display(df_cleaned.head(10))

# Next Steps: Similarity & Deduplication
```

```
Some weights of the model checkpoint at dslim/bert-base-NER were not used when
initializing BertForTokenClassification: ['bert.pooler.dense.bias',
'bert.pooler.dense.weight']
- This IS expected if you are initializing BertForTokenClassification from the
checkpoint of a model trained on another task or with another architecture (e.g.
initializing a BertForSequenceClassification model from a BertForPreTraining
model).
- This IS NOT expected if you are initializing BertForTokenClassification from
the checkpoint of a model that you expect to be exactly identical (initializing
a BertForSequenceClassification model from a BertForSequenceClassification
model).
Device set to use cuda:0
```

```
Using Dask to load large dataset.
Raw DataFrame Loaded:
Shape: (33446, 75)
Large dataset detected. Using Dask map_partitions.

Cleaned & Enriched DataFrame (Sample):
Shape: (33446, 76)
```

```
      company_name company_legal_names  \
0     owensliquors                <NA>
1      clubtarneit                <NA>
```

```
2          aaaauto                <NA>
3         gisinger       Gisinger GmbH
4      kasana life                <NA>
5  bammakeupstudio                <NA>
6          tescoma                <NA>
7    happyweddings                <NA>
8     dentalplanet                <NA>
9      kdrakephoto                <NA>

                           company_commercial_names main_country_code  \
0                                        Owens Liquors                US
1                                         Club Tarneit                AU
2                           AAA Auto Otrokovice Zlín                  CZ
3                                                <NA>                 DE
4                                         Kasana Life                US
5                           BAM BROW & MAKEUP STUDIO                 AU
6                                              Tescoma                HU
7  Happyweddings | No.1 Matrimony Trivandrum Kera…                   IN
8                                Dental Planet Manukau                NZ
9                           Drake Design Photography                  US

    main_country        main_region      main_city_district  \
0  united states     south carolina                    None
1      australia           victoria                 tarneit
2        czechia               zlín  kvítkovice u otrokovic
3        germany  baden-württemberg                    None
4  united states        connecticut                    None
5      australia  western australia                    None
6        hungary           budapest                    None
7          india             kerala                    None
8    new zealand           auckland                    None
9  united states              texas                    None

             main_city main_postcode                          main_street  …  \
0       pawleys island         29585                       ocean highway  …
1      city of wyndham          3029                                None  …
2           otrokovice        765 02                              zlínská  …
3  ühlingen-birkendorf         79777                        berauer straße  …
4           litchfield         06759                                None  …
5             mandurah          6201                                None  …
6             budapest          1222                       nagytétényi út  …
7   thiruvananthapuram        695025  medical college - chalakkuzhy road  …
8             auckland          2104                     bakerfield place  …
9              lubbock         79416                     county road 6430  …

                            generated_business_tags  status domains  \
0       Retail Trade | Liquor Stores | Wine & Liquor  Active    <NA>
1                                               <NA>  Active    <NA>
```

```
2  In-store Shopping | Investment Management Serv…  Active    <NA>
3                                                <NA>  Active    <NA>
4                                                <NA>  Active    <NA>
5                                                <NA>  Active    <NA>
6  Home Furnishings Retailer | Kitchenchen Supply…  Active    <NA>
7  Event Planning Services | Wedding Planning Ser…  Active    <NA>
8        Healthcare Services | Ddental Care Services  Active    <NA>
9  Wheelchair Access | Portrait Studio | Photogra…  Active    <NA>

  all_domains revenue revenue_type employee_count employee_count_type  \
0        <NA>     NaN         <NA>            NaN                <NA>
1        <NA>     NaN         <NA>            9.0           extracted
2        <NA>     NaN         <NA>            NaN                <NA>
3        <NA>     NaN         <NA>            NaN                <NA>
4        <NA>     NaN         <NA>            NaN                <NA>
5        <NA>     NaN         <NA>            NaN                <NA>
6        <NA>     NaN         <NA>            NaN                <NA>
7        <NA>     NaN         <NA>            NaN                <NA>
8        <NA>     NaN         <NA>            NaN                <NA>
9        <NA>     NaN         <NA>            NaN                <NA>

  inbound_links_count                                  activity_enriched
0                <NA>  445320, 47.91 | 47.25 | 47.81 | 47.99, 4722 | …
1                <NA>                                   events & service
2                <NA>  441120, 45.11 | 45.19, 4510, 5521, automobile …
3                <NA>                                              None
4                <NA>                                              None
5                <NA>                                              None
6                <NA>  449129, 47.89 | 47.59 | 47.54 | 47.76 | 47.77 …
7                <NA>                                 happyweddings. com
8                <NA>  621210, 8021, 86.22 | 86.23 | 86.21, 8620, den…
9                <NA>  541921, 7221, 74.2, 7420, photographers & phot…

[10 rows x 76 columns]
```

[ ]:
```python
# number of non-null values in each column
non_null_counts = df_cleaned.notna().sum(axis=0)

for column_name, count in non_null_counts.items():
    print(f"{column_name}: {count}")
```

```
company_name: 33397
company_legal_names: 6890
company_commercial_names: 28121
main_country_code: 31415
main_country: 31509
main_region: 30112
main_city_district: 5979
```

main_city: 29602
main_postcode: 23820
main_street: 19979
main_street_number: 17034
main_latitude: 17031
main_longitude: 17031
main_address_raw_text: 27980
locations: 31415
num_locations: 19110
company_type: 19735
year_founded: 5027
lnk_year_founded: 2142
short_description: 18702
long_description: 11731
business_tags: 9249
business_model: 19798
product_type: 19798
naics_vertical: 18263
naics_2022_primary_code: 18048
naics_2022_primary_label: 18048
naics_2022_secondary_codes: 244
naics_2022_secondary_labels: 244
main_business_category: 23615
main_industry: 23615
main_sector: 19798
primary_phone: 22799
phone_numbers: 22799
primary_email: 6508
emails: 3316
other_emails: 578
website_url: 31893
website_domain: 31893
website_tld: 31893
website_language_code: 6104
facebook_url: 11282
twitter_url: 2755
instagram_url: 7014
linkedin_url: 10200
ios_app_url: 123
android_app_url: 138
youtube_url: 1142
tiktok_url: 0
alexa_rank: 0
sics_codified_industry: 5661
sics_codified_industry_code: 5661
sics_codified_subsector: 5661
sics_codified_subsector_code: 5661
sics_codified_sector: 5661

```
sics_codified_sector_code: 5661
sic_codes: 18048
sic_labels: 18048
isic_v4_codes: 18048
isic_v4_labels: 18048
nace_rev2_codes: 18048
nace_rev2_labels: 18048
created_at: 33403
last_updated_at: 33403
website_number_of_pages: 6151
generated_description: 19468
generated_business_tags: 19445
status: 33446
domains: 11592
all_domains: 11592
revenue: 7229
revenue_type: 7229
employee_count: 8723
employee_count_type: 8723
inbound_links_count: 6151
activity_enriched: 23575
```

[ ]:
```python
non_null_counts = df_cleaned.notna().sum(axis=0)

for column_name, count in non_null_counts.items():
    print(f"{column_name}: {count}")
```

```
company_name: 33397
company_legal_names: 6890
company_commercial_names: 28121
main_country_code: 31415
main_country: 31509
main_region: 30112
main_city_district: 5979
main_city: 29602
main_postcode: 23820
main_street: 19979
main_street_number: 17034
main_latitude: 17031
main_longitude: 17031
main_address_raw_text: 27980
locations: 31415
num_locations: 19110
company_type: 19735
year_founded: 5027
lnk_year_founded: 2142
short_description: 18702
long_description: 11731
```

business_tags: 9249
business_model: 19798
product_type: 19798
naics_vertical: 18263
naics_2022_primary_code: 18048
naics_2022_primary_label: 18048
naics_2022_secondary_codes: 244
naics_2022_secondary_labels: 244
main_business_category: 23615
main_industry: 23615
main_sector: 19798
primary_phone: 22799
phone_numbers: 22799
primary_email: 6508
emails: 3316
other_emails: 578
website_url: 31893
website_domain: 31893
website_tld: 31893
website_language_code: 6104
facebook_url: 11282
twitter_url: 2755
instagram_url: 7014
linkedin_url: 10200
ios_app_url: 123
android_app_url: 138
youtube_url: 1142
tiktok_url: 0
alexa_rank: 0
sics_codified_industry: 5661
sics_codified_industry_code: 5661
sics_codified_subsector: 5661
sics_codified_subsector_code: 5661
sics_codified_sector: 5661
sics_codified_sector_code: 5661
sic_codes: 18048
sic_labels: 18048
isic_v4_codes: 18048
isic_v4_labels: 18048
nace_rev2_codes: 18048
nace_rev2_labels: 18048
created_at: 33403
last_updated_at: 33403
website_number_of_pages: 6151
generated_description: 19468
generated_business_tags: 19445
status: 33446
domains: 11592

```
all_domains: 11592
revenue: 7229
revenue_type: 7229
employee_count: 8723
employee_count_type: 8723
inbound_links_count: 6151
activity_enriched: 23575
```

[76]:
```
df_raw.head(10)
df_cleanedd=df_cleaned
df_cleanedd
```

[76]:

	company_name	company_legal_names	company_commercial_names	main_country_code	main_country	main_region	main_city_district
0	owensliquors	<NA>	Owens Liquors	US	united states	south carolina	None
1	clubtarneit	<NA>	Club Tarneit	AU	australia	victoria	tarneit
2	aaaauto	<NA>	AAA Auto Otrokovice Zlín	CZ	czechia	zlín	kvítkovice u otrokovic
3	gisinger	Gisinger GmbH	<NA>	DE	germany	baden-württemberg	None
4	kasana life	<NA>	Kasana Life	US	united states	connecticut	None
…	…	…	…	…	…	…	…
33441	gemspright	Gem Spright Electricals & Automation Pvt Ltd.	<NA>	<NA>	None	None	None
33442	sport4u	<NA>	SPORT4U parduotuvė - UAB Vitaga ir ko	LT	lithuania	kaunas county	None
33443	city-sightseeing	<NA>	City Sightseeing Norwich	GB			
33444	trimbakeshwarmandir	<NA>	Trimbakeshwar Jyotirling Temple	IN			
33445	67thdc	<NA>	Grand Blanc District Court	US			

```
33443  united kingdom            england                    None
33444           india        maharashtra                    None
33445   united states           michigan                    None

                main_city main_postcode           main_street  …  \
0           pawleys island         29585         ocean highway  …
1          city of wyndham          3029                  None  …
2               otrokovice        765 02              zlínská  …
3      ühlingen-birkendorf         79777        berauer straße  …
4               litchfield         06759                  None  …
…                    …           …                     …  …
33441                None          None                  None  …
33442              kaunas      lt-51189          pramonės pr.  …
33443              cromer          None                  None  …
33444              nashik        422003          college road  …
33445          grand blanc         48439  south saginaw street  …

                                generated_business_tags  status  \
0           Retail Trade | Liquor Stores | Wine & Liquor  Active
1                                                   <NA>  Active
2      In-store Shopping | Investment Management Serv…  Active
3                                                   <NA>  Active
4                                                   <NA>  Active
…                                                    …     …
33441                                               <NA>  Active
33442  In-store Pickup | In-store Shopping | Wheelcha…  Active
33443  Open Bus Transportation | Transportation Services  Active
33444  Cleanliness Services | H Accommodation Service…  Active
33445  Governmental | Non-profit Organization | Acces…  Active

                   domains           all_domains revenue revenue_type  \
0                     <NA>                  <NA>     NaN         <NA>
1                     <NA>                  <NA>     NaN         <NA>
2                     <NA>                  <NA>     NaN         <NA>
3                     <NA>                  <NA>     NaN         <NA>
4                     <NA>                  <NA>     NaN         <NA>
…                      …                     …     …           …
33441                 <NA>                  <NA>     NaN         <NA>
33442                 <NA>                  <NA>     NaN         <NA>
33443  city-sightseeing.com  city-sightseeing.com     NaN         <NA>
33444                 <NA>                  <NA>     NaN         <NA>
33445                 <NA>                  <NA>     NaN         <NA>

      employee_count employee_count_type inbound_links_count  \
0                NaN                <NA>                <NA>
1                9.0           extracted                <NA>
2                NaN                <NA>                <NA>
```

```
3                NaN                <NA>                <NA>
4                NaN                <NA>                <NA>
…                …                  …                   …
33441            NaN                <NA>                <NA>
33442            NaN                <NA>                <NA>
33443            NaN                <NA>                <NA>
33444            NaN                <NA>                <NA>
33445            NaN                <NA>                <NA>

                                        activity_enriched
0      445320, 47.91 | 47.25 | 47.81 | 47.99, 4722 | …
1                                         events & service
2      441120, 45.11 | 45.19, 4510, 5521, automobile …
3                                                      None
4                                                      None
…                                                      …
33441                                                  None
33442  459110, 47.91 | 47.64 | 47.76 | 47.77 | 47.78 …
33443  4725, 561520, 79.12, 7912, panoramic views, to…
33444  813110, 8661, 94.91, 9491, activities of relig…
33445                                                  None

[33446 rows x 76 columns]
```

[52]:
```
df_cleaned=df_cleanedd
```

[ ]:
```
import pandas as pd
import numpy as np
from rapidfuzz import fuzz

try:
    df_cleaned
except NameError:
    #  mini DataFrame
    data = {
        "id": [1, 2, 3, 4, 5, 6],
        "company_name": [
            "Acme Inc", "Acme Incorporated", "Global Tech",
            "Global Tech Solutions", "FooBar LLC", "FooBar LTD"
        ],
        "main_country": ["US", "US", "US", "US", None, None],
        "revenue": [100000, 110000, 500000, 510000, None, None],
        "employee_count": [50, 52, 200, 220, None, 210],
        "website_domain": [
            "acme.com", "acme.com", "globaltech.com",
            "globaltech.com", "foobar.com", "foobar.com"
        ],
```

```
        "facebook_url": [
            None, None, "facebook.com/GlobalTech",
            "facebook.com/globaltech", "facebook.com/foobar", "facebook.com/
 ↪foobar"
        ],
        "twitter_url": [
            None, None, None, "twitter.com/globaltech", None, None
        ]
    }
    df_cleaned = pd.DataFrame(data)

print("df_cleaned sample:")
display(df_cleaned)

# Split known vs unknown country
df_known = df_cleaned[~df_cleaned["main_country"].isna()].copy()
df_unknown = df_cleaned[df_cleaned["main_country"].isna()].copy()

# Group by country, also handle missing as "MISSING_COUNTRY"
groups = {}
if not df_known.empty:
    for country, subdf in df_known.groupby("main_country"):
        groups[country] = subdf

if not df_unknown.empty:
    groups["MISSING_COUNTRY"] = df_unknown

duplicates = []

# List of social media columns for partial scoring
SOCIAL_COLS = [
    "facebook_url", "twitter_url", "instagram_url",
    "linkedin_url", "ios_app_url", "android_app_url",
    "youtube_url", "tiktok_url"
]


def within_10pct(valA, valB):

    if valA is None or valB is None:
        return False
    # must be numeric
    try:
        valA = float(valA)
        valB = float(valB)
    except:
        return False
```

```
    # check ratio
    if valA == 0 or valB == 0:
        return False
    ratio = valA / valB
    return 0.7 <= ratio <= 1.3


def compute_pair_score(rowA, rowB):

    points = 0

    # 1. Name similarity up to 2 points
    nameA = rowA.get("company_name") or ""
    nameB = rowB.get("company_name") or ""
    ratio = fuzz.ratio(nameA, nameB)  # [0..100]
    if ratio >= 80:
        points += 1.5
    elif ratio >= 50:
        points += 1
    elif ratio >= 30:
        points += 0.5
    elif ratio >= 10:
        points -= 1.0
    else:
        points -= 3.0

    # 2. Revenue ±10% => +1
    revA = rowA.get("revenue")
    revB = rowB.get("revenue")
    if within_10pct(revA, revB):
        points += 0.5

    # 3. Employee_count ±10% => +1
    empA = rowA.get("employee_count")
    empB = rowB.get("employee_count")
    if within_10pct(empA, empB):
        points += 0.5

    # 4. Website domain exact => +2 if match and not empty
    domA = rowA.get("website_domain") or ""
    domB = rowB.get("website_domain") or ""
    if domA and domB and domA.lower() == domB.lower():
        points += 2

    # 5. Social media => +2 each if exact
    for col in SOCIAL_COLS:
        valA = rowA.get(col) or ""
```

```
        valB = rowB.get(col) or ""
        if valA and valB and valA.lower() == valB.lower():
            points += 1



    #6. Different city
    cityA = rowA.get("main_city")
    cityB = rowB.get("main_city")
    if cityA and cityB and fuzz.ratio(cityA, cityB) < 80:
        points -= 2

    # 7. Different country
    countryA = rowA.get("main_country")
    countryB = rowB.get("main_country")
    if countryA and countryB and countryA.lower() != countryB.lower():
        points -= 2

    # 8. Different region
    regionA = rowA.get("main_region")
    regionB = rowB.get("main_region")
    if regionA and regionB and fuzz.ratio(regionA, regionB) < 80:
        points -= 1
    return points


# Evaluate pairs within each block
for block_key, block_df in groups.items():
    if len(block_df) < 2:
        continue  # no pairs
    records = block_df.to_dict(orient="index")
    idx_list = list(records.keys())

    for i in range(len(idx_list)):
        for j in range(i+1, len(idx_list)):
            idxA = idx_list[i]
            idxB = idx_list[j]
            rowA = records[idxA]
            rowB = records[idxB]

            score = compute_pair_score(rowA, rowB)
            # threshold
            if score >= 4:
                duplicates.append((idxA, idxB, score, block_key))

# Summarize duplicates
df_duplicates = pd.DataFrame(
```

```
    duplicates, columns=["idxA", "idxB", "score", "country_block"])
df_duplicates.sort_values("score", ascending=False, inplace=True)

print(f"Found {len(df_duplicates)} potential duplicates with score >= 4:")
display(df_duplicates)
```

df_cleaned sample:

```
             company_name                              company_legal_names  \
0             owensliquors                                             <NA>
1              clubtarneit                                             <NA>
2                  aaaauto                                             <NA>
3                 gisinger                                  Gisinger GmbH
4              kasana life                                             <NA>
…                     …                                                …
33441           gemspright  Gem Spright Electricals & Automation Pvt Ltd.
33442              sport4u                                             <NA>
33443    city-sightseeing                                              <NA>
33444  trimbakeshwarmandir                                             <NA>
33445              67thdc                                              <NA>

                  company_commercial_names main_country_code  \
0                            Owens Liquors                US
1                             Club Tarneit                AU
2                  AAA Auto Otrokovice Zlín                CZ
3                                     <NA>                DE
4                              Kasana Life                US
…                                      …                …
33441                                 <NA>              <NA>
33442  SPORT4U parduotuvė - UAB Vitaga ir ko                LT
33443              City Sightseeing Norwich                GB
33444       Trimbakeshwar Jyotirling Temple                IN
33445            Grand Blanc District Court                US

         main_country        main_region      main_city_district  \
0       united states     south carolina                    None
1           australia           victoria                 tarneit
2             czechia               zlín  kvítkovice u otrokovic
3             germany  baden-württemberg                    None
4       united states        connecticut                    None
…                 …                  …                       …
33441           None               None                    None
33442      lithuania      kaunas county                    None
33443  united kingdom            england                    None
33444          india        maharashtra                    None
33445   united states           michigan                    None

                main_city main_postcode           main_street  …  \
```

```
0           pawleys island         29585         ocean highway  …
1          city of wyndham          3029                  None  …
2               otrokovice        765 02               zlínská  …
3      ühlingen-birkendorf         79777        berauer straße  …
4                litchfield         06759                  None  …
…                     …           …                    …  …
33441                 None          None                  None  …
33442               kaunas      lt-51189          pramonės pr.  …
33443               cromer          None                  None  …
33444               nashik        422003          college road  …
33445          grand blanc         48439  south saginaw street  …

                                generated_business_tags  status  \
0           Retail Trade | Liquor Stores | Wine & Liquor  Active
1                                                   <NA>  Active
2      In-store Shopping | Investment Management Serv…  Active
3                                                   <NA>  Active
4                                                   <NA>  Active
…                                                    …     …
33441                                               <NA>  Active
33442  In-store Pickup | In-store Shopping | Wheelcha…  Active
33443  Open Bus Transportation | Transportation Services  Active
33444  Cleanliness Services | H Accommodation Service…  Active
33445  Governmental | Non-profit Organization | Acces…  Active

                    domains           all_domains revenue revenue_type  \
0                      <NA>                  <NA>     NaN         <NA>
1                      <NA>                  <NA>     NaN         <NA>
2                      <NA>                  <NA>     NaN         <NA>
3                      <NA>                  <NA>     NaN         <NA>
4                      <NA>                  <NA>     NaN         <NA>
…                       …                     …     …           …
33441                  <NA>                  <NA>     NaN         <NA>
33442                  <NA>                  <NA>     NaN         <NA>
33443  city-sightseeing.com  city-sightseeing.com     NaN         <NA>
33444                  <NA>                  <NA>     NaN         <NA>
33445                  <NA>                  <NA>     NaN         <NA>

      employee_count employee_count_type inbound_links_count  \
0                NaN                <NA>                <NA>
1                9.0           extracted                <NA>
2                NaN                <NA>                <NA>
3                NaN                <NA>                <NA>
4                NaN                <NA>                <NA>
…               …                   …                   …
33441            NaN                <NA>                <NA>
33442            NaN                <NA>                <NA>
33443            NaN                <NA>                <NA>
```

```
33444            NaN                <NA>                <NA>
33445            NaN                <NA>                <NA>

                                        activity_enriched
0      445320, 47.91 | 47.25 | 47.81 | 47.99, 4722 | …
1                                        events & service
2      441120, 45.11 | 45.19, 4510, 5521, automobile …
3                                                    None
4                                                    None
…                                                     …
33441                                                None
33442  459110, 47.91 | 47.64 | 47.76 | 47.77 | 47.78 …
33443  4725, 561520, 79.12, 7912, panoramic views, to…
33444  813110, 8661, 94.91, 9491, activities of relig…
33445                                                None

[33446 rows x 76 columns]
```

```
Found 13477 potential duplicates with score >= 4:

        idxA   idxB  score         country_block
4866    1107  12539    9.0               malaysia
7147    1254  32388    8.5         united kingdom
1915   13838  25528    8.5                 canada
5719   21693  28759    8.5            philippines
11422  11867  29000    8.5          united states
…        …    …        …                    …
10669   8041  31078    4.0          united states
6993    4709   8655    4.0  united arab emirates
10029   5141  30412    4.0          united states
13116  27133  30122    4.0          united states
7408    4109  21841    4.0         united kingdom

[13477 rows x 4 columns]
```

[79]:
```python
# PART 3
import pandas as pd
import networkx as nx  # pip install networkx
import pyarrow as pa   # for parquet
import pyarrow.parquet as pq


try:
    df_cleaned
    df_duplicates
except NameError:
    # Minimal example
    df_cleaned = pd.DataFrame({
```

```
        "id": [1, 2, 3, 4],
        "company_name": ["Acme Inc", "Acme Incorporated", "Global Tech",␣
 ↪"Global Tech Solutions"],
        "main_country": ["US", "US", "US", "US"],
        "some_text_col": ["short text", "much longer text about Acme", None,␣
 ↪"Global Tech Solutions Inc."],
        "some_num_col": [10, 20, 30, 25],
    }).set_index("id", drop=False)

    # Suppose we found duplicates
    df_duplicates = pd.DataFrame({
        "idxA": [1, 3],
        "idxB": [2, 4],
        "score": [5, 6],
        "country_block": ["US", "US"]
    })

print("df_cleaned:")
display(df_cleaned.head())
print("\ndf_duplicates:")
display(df_duplicates)

# 1. Build a graph of duplicates => connected components
# Create a graph where each row is a node, each duplicate pair is an edge
G = nx.Graph()
all_indices = df_cleaned.index.tolist()
G.add_nodes_from(all_indices)

for row in df_duplicates.itertuples():
    idxA = row.idxA
    idxB = row.idxB
    G.add_edge(idxA, idxB)  # no direction needed for duplicates

# Find connected components
connected_components = list(nx.connected_components(G))
print(f"Found {len(connected_components)} connected components (clusters).")

# 2. Merge each cluster

def merge_cluster(cluster_indices, df):
    """
    Merge all rows in cluster_indices into one 'golden record'.
    Heuristics:
      - For text cols, pick the row with the 'longest string'.
      - For numeric, pick the largest or first non-null (choose your logic).
      - Otherwise pick first non-null.
    """
```

```
    subset = df.loc[cluster_indices]

    merged_row = {}

    representative_idx = min(cluster_indices)

    for col in df.columns:
        col_values = subset[col].dropna().tolist()  # remove null
        if len(col_values) == 0:
            # everything is null
            merged_row[col] = None
            continue

        # Check dtypes
        if subset[col].dtype == object:
            # treat as text, pick the 'longest'
            best_val = max(col_values, key=lambda x: len(str(x)))
            merged_row[col] = best_val
        elif pd.api.types.is_numeric_dtype(subset[col].dtype):
            # pick the largest numeric
            numeric_vals = [v for v in col_values if pd.notna(v)]
            if numeric_vals:
                best_val = max(numeric_vals)
                merged_row[col] = best_val
            else:
                merged_row[col] = None
        else:
            # fallback: pick first
            merged_row[col] = col_values[0]

    # choose an ID for the merged row
    merged_row["id"] = representative_idx

    return merged_row


# accumulate merged rows in a list
merged_records = []

for comp in connected_components:
    if len(comp) == 1:
        # single node => no duplicates => just copy the row
        idx = list(comp)[0]
        # convert the row to dict
        row_dict = df_cleaned.loc[idx].to_dict()
        merged_records.append(row_dict)
    else:
```

```
        # merge the cluster
        merged = merge_cluster(list(comp), df_cleaned)
        merged_records.append(merged)

df_merged = pd.DataFrame(merged_records)

print("\nMerged (Golden) Records:")
display(df_merged)

df_merged.set_index("id", drop=True, inplace=True)

# 3. Save to final_data.parquet
df_merged.to_parquet("final_data.parquet")
print("\nSaved final merged data to final_data.parquet")
```

df_cleaned:

```
   company_name company_legal_names  company_commercial_names  \
0  owensliquors                <NA>             Owens Liquors
1   clubtarneit                <NA>              Club Tarneit
2       aaaauto                <NA>  AAA Auto Otrokovice Zlín
3      gisinger       Gisinger GmbH                      <NA>
4   kasana life                <NA>               Kasana Life

  main_country_code   main_country        main_region      main_city_district  \
0                US  united states     south carolina                    None
1                AU      australia           victoria                 tarneit
2                CZ        czechia               zlín  kvítkovice u otrokovic
3                DE        germany  baden-württemberg                    None
4                US  united states        connecticut                    None

            main_city main_postcode     main_street  …  \
0       pawleys island         29585   ocean highway  …
1      city of wyndham          3029            None  …
2           otrokovice        765 02         zlínská  …
3  ühlingen-birkendorf         79777  berauer straße  …
4           litchfield         06759            None  …

                           generated_business_tags  status domains  \
0       Retail Trade | Liquor Stores | Wine & Liquor  Active    <NA>
1                                              <NA>  Active    <NA>
2  In-store Shopping | Investment Management Serv…  Active    <NA>
3                                              <NA>  Active    <NA>
4                                              <NA>  Active    <NA>

  all_domains revenue revenue_type employee_count employee_count_type  \
0        <NA>     NaN         <NA>            NaN                <NA>
1        <NA>     NaN         <NA>            9.0           extracted
```

```
2        <NA>     NaN         <NA>            NaN                <NA>
3        <NA>     NaN         <NA>            NaN                <NA>
4        <NA>     NaN         <NA>            NaN                <NA>

  inbound_links_count                                    activity_enriched
0                <NA>  445320, 47.91 | 47.25 | 47.81 | 47.99, 4722 | …
1                <NA>                                    events & service
2                <NA>  441120, 45.11 | 45.19, 4510, 5521, automobile …
3                <NA>                                                None
4                <NA>                                                None

[5 rows x 76 columns]


df_duplicates:

        idxA   idxB  score         country_block
4866    1107  12539    9.0               malaysia
7147    1254  32388    8.5         united kingdom
1915   13838  25528    8.5                 canada
5719   21693  28759    8.5            philippines
11422  11867  29000    8.5          united states
…      …    …    …                    …
10669   8041  31078    4.0          united states
6993    4709   8655    4.0  united arab emirates
10029   5141  30412    4.0          united states
13116  27133  30122    4.0          united states
7408    4109  21841    4.0         united kingdom

[13477 rows x 4 columns]

Found 24445 connected components (clusters).

Merged (Golden) Records:

              company_name company_legal_names  \
0              owensliquors                None
1               clubtarneit                None
2                   aaaauto                None
3                  gisinger       Gisinger GmbH
4               kasana life                None
…                      …                 …
24440       westbrookmaine                None
24441              sport4u                None
24442     city-sightseeing                None
24443  trimbakeshwarmandir                None
24444               67thdc                None

                   company_commercial_names main_country_code  \
0                             Owens Liquors                US

                                                                  28
```

```
1                              Club Tarneit                AU
2                  AAA Auto Otrokovice Zlín                CZ
3                                      None                DE
4                               Kasana Life                US
…                                       …               …
24440                            Westbrook                US
24441  SPORT4U parduotuvė - UAB Vitaga ir ko                LT
24442              City Sightseeing Norwich                GB
24443       Trimbakeshwar Jyotirling Temple                IN
24444            Grand Blanc District Court                US

         main_country        main_region     main_city_district  \
0       united states     south carolina                   None
1           australia           victoria                tarneit
2             czechia               zlín  kvítkovice u otrokovic
3             germany  baden-württemberg                   None
4       united states        connecticut                   None
…                 …                  …                      …
24440   united states              maine                   None
24441       lithuania      kaunas county                   None
24442  united kingdom            england                   None
24443           india        maharashtra                   None
24444   united states           michigan                   None

                main_city main_postcode          main_street  …  status  \
0          pawleys island         29585        ocean highway  …  Active
1         city of wyndham          3029                 None  …  Active
2              otrokovice        765 02              zlínská  …  Active
3     ühlingen-birkendorf         79777       berauer straße  …  Active
4              litchfield         06759                 None  …  Active
…                     …             …                    …  …     …
24440           westbrook         04092       hemphill drive  …  Active
24441              kaunas      lt-51189         pramonės pr.  …  Active
24442              cromer          None                 None  …  Active
24443              nashik        422003         college road  …  Active
24444         grand blanc         48439  south saginaw street  …  Active

                   domains           all_domains revenue revenue_type  \
0                     None                  None     NaN         None
1                     None                  None     NaN         None
2                     None                  None     NaN         None
3                     None                  None     NaN         None
4                     None                  None     NaN         None
…                      …                    …     …          …
24440                 None                  None     NaN         None
24441                 None                  None     NaN         None
24442  city-sightseeing.com  city-sightseeing.com     NaN         None
24443                 None                  None     NaN         None
```

```
24444                  None                    None     NaN         None

      employee_count employee_count_type inbound_links_count  \
0                NaN                None                None
1                9.0           extracted                None
2                NaN                None                None
3                NaN                None                None
4                NaN                None                None
…              …                  …                   …
24440            NaN                None                None
24441            NaN                None                None
24442            NaN                None                None
24443            NaN                None                None
24444            NaN                None                None

                                      activity_enriched  id
0      445320, 47.91 | 47.25 | 47.81 | 47.99, 4722 | … NaN
1                                       events & service NaN
2      441120, 45.11 | 45.19, 4510, 5521, automobile … NaN
3                                                  None NaN
4                                                  None NaN
…                                                  …  ..
24440  ##s, food & drink, health & beauty, westbrook … NaN
24441  459110, 47.91 | 47.64 | 47.76 | 47.77 | 47.78 … NaN
24442  4725, 561520, 79.12, 7912, panoramic views, to… NaN
24443  813110, 8661, 94.91, 9491, activities of relig… NaN
24444                                              None NaN

[24445 rows x 77 columns]


Saved final merged data to final_data.parquet
```

[ ]:
```python
# Verification snippet: print original rows of the first non-trivial cluster
for comp in connected_components:
    if len(comp) > 1:
        print(f"\n--- Original Rows from Cluster {comp} ---")
        display(df_cleaned.loc[list(comp)])
```