A2: H1B Visas

Manual to Normalize H1B Visas Files and Upload it to MySQL Server

Hult International Business School

Data Management & SQL - DAT-5486 - BMBAND1

Team 15

Alvaro Ronquillo
Karan Padhiyar
Melika Saeedi

Professor Luis Escamilla

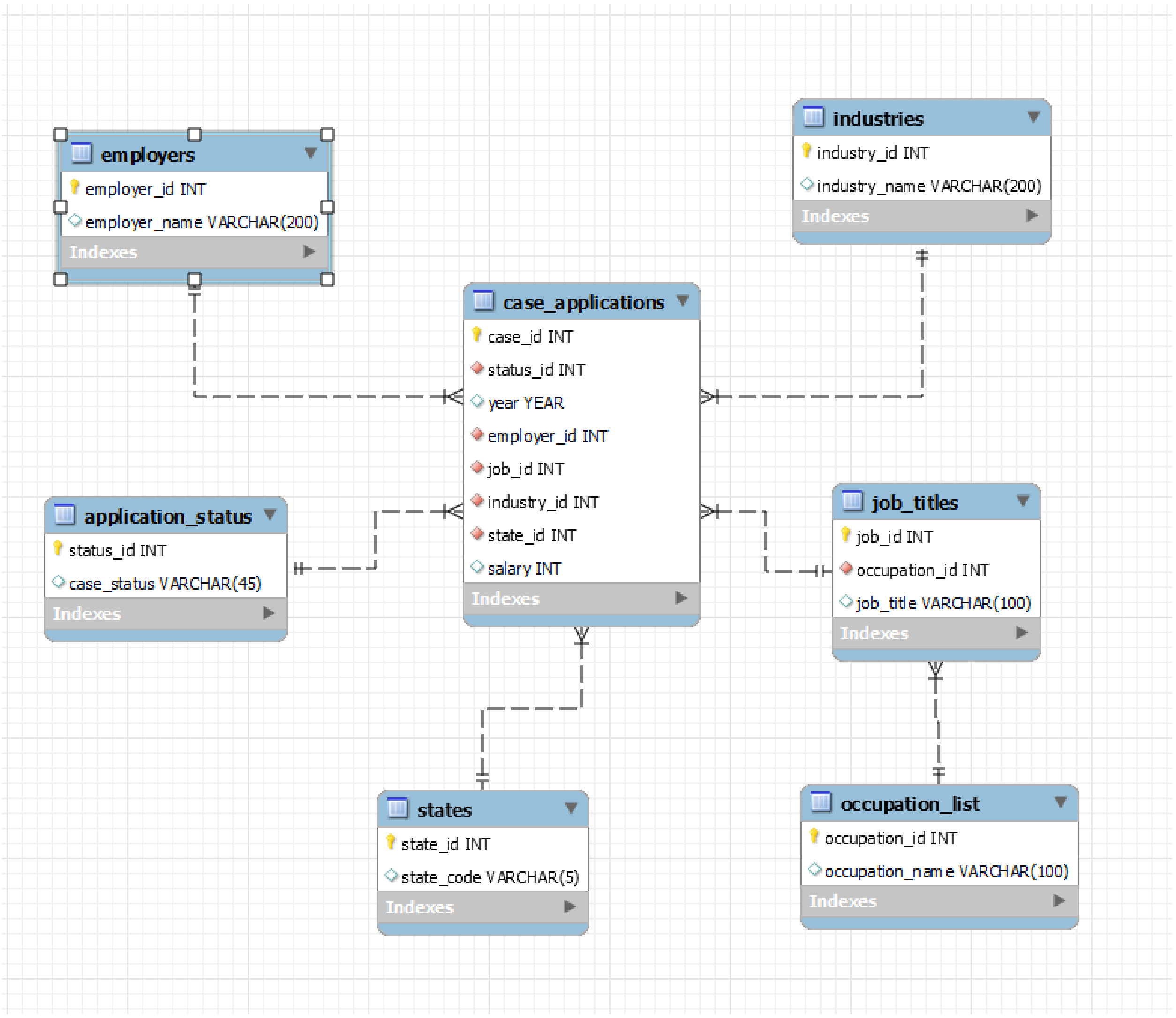
November 9, 2022

Creating the Scheme and EER Diagram

Before cleaning the data, we must first determine which data is relevant for a h1b candidate searching for a job in the US.

After that we start creating the schema and tables with the column name and data type.

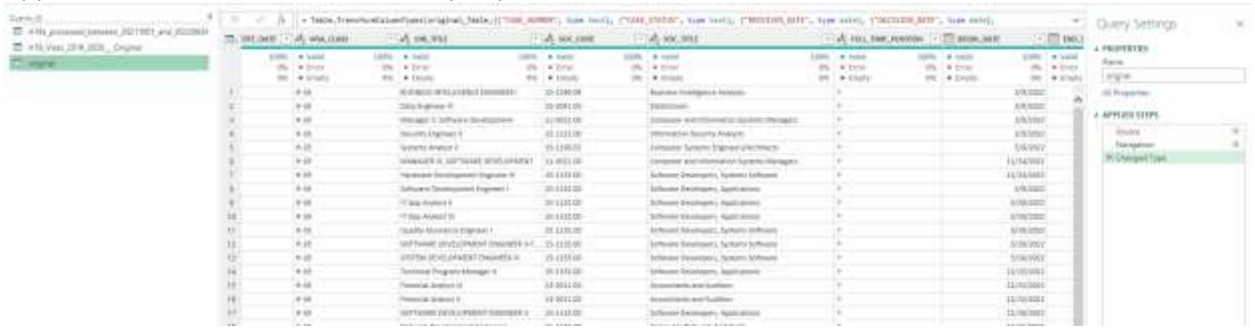
The final schema looks like this:



Cleaning the Data in Power Query

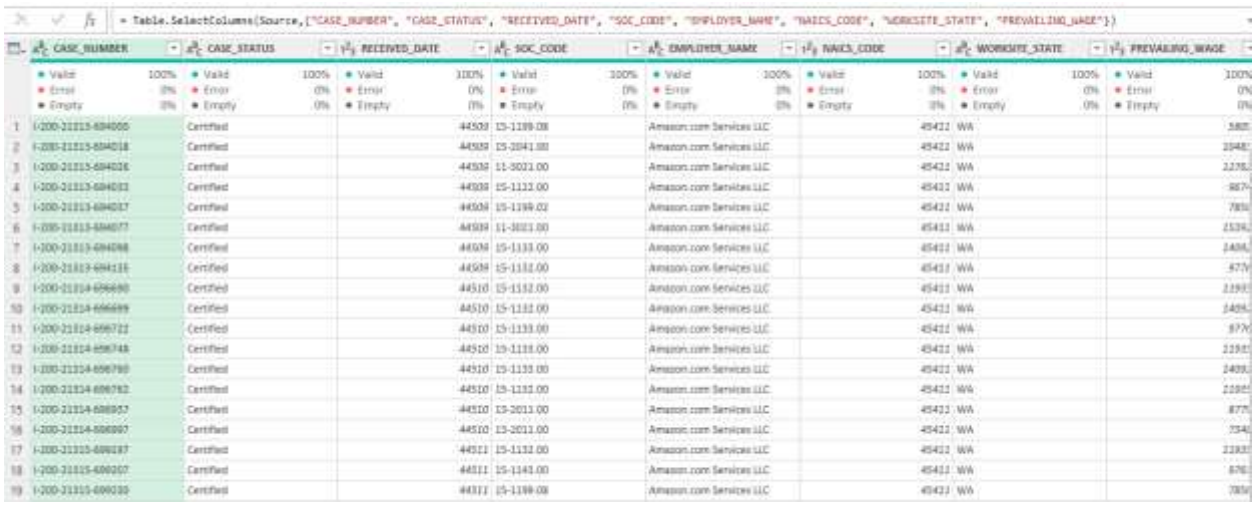
The first step for this assignment was to figure out the kind of questions an H1B candidate would have when searching for jobs.

1. Append the H1B tables and create a query with all the information needed.



2. The amount of data was too big for just using one file, so we saved the complete file and an extra copy of just the "original" query so we can work with it and the operations wouldn't take so long.

3. Reference the original query and name it "case_applications". This is going to be our main table.
4. Remove the Columns that are not needed and stay with and rearrange the columns in the same way as your table created in the SQL server:
   I. Case Number
   II. Case Status
   III. Received Date
   IV. SOC Code
   V. Employer Name
   VI. NAICS Code
   VII. Worksite State
   VIII. Prevailing Wage



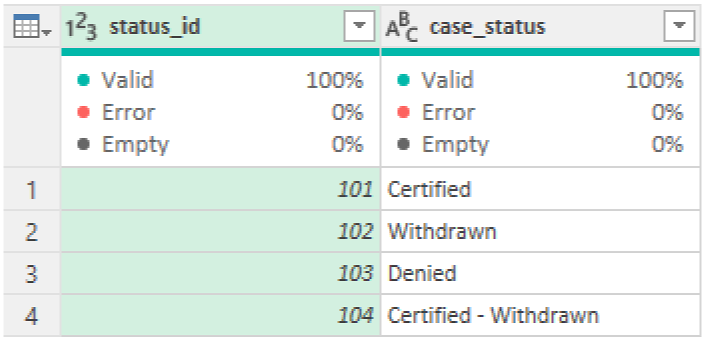
```
= Table.SelectColumns(Source,{"CASE_NUMBER", "CASE_STATUS", "RECEIVED_DATE", "SOC_CODE", "EMPLOYER_NAME", "NAICS_CODE", "WORKSITE_STATE", "PREVAILING_WAGE"})
```

	CASE_NUMBER	CASE_STATUS	RECEIVED_DATE	SOC_CODE	EMPLOYER_NAME	NAICS_CODE	WORKSITE_STATE	PREVAILING_WAGE
	● Valid 100% ● Error 0% ● Empty 0%	● Valid 100% ● Error 0% ● Empty 0%	● Valid 100% ● Error 0% ● Empty 0%	● Valid 100% ● Error 0% ● Empty 0%	● Valid 100% ● Error 0% ● Empty 0%	● Valid 100% ● Error 0% ● Empty 0%	● Valid 100% ● Error 0% ● Empty 0%	● Valid 100% ● Error 0% ● Empty 0%
1	I-200-21313-694000	Certified	44509	15-1199.08	Amazon.com Services LLC	45422	WA	3805
2	I-200-21313-694018	Certified	44509	15-2041.00	Amazon.com Services LLC	45422	WA	2048
3	I-200-21313-694026	Certified	44509	11-3021.00	Amazon.com Services LLC	45422	WA	2278
4	I-200-21313-694033	Certified	44509	15-1122.00	Amazon.com Services LLC	45422	WA	9874
5	I-200-21313-694037	Certified	44509	15-1199.02	Amazon.com Services LLC	45422	WA	785
6	I-200-21313-694077	Certified	44509	11-3021.00	Amazon.com Services LLC	45411	WA	2534
7	I-200-21313-694088	Certified	44509	15-1133.00	Amazon.com Services LLC	45411	WA	2408
8	I-200-21313-694135	Certified	44509	15-1132.00	Amazon.com Services LLC	45411	WA	877
9	I-200-21314-696690	Certified	44510	15-1132.00	Amazon.com Services LLC	45411	WA	2193
10	I-200-21314-696699	Certified	44510	15-1132.00	Amazon.com Services LLC	45411	WA	2409
11	I-200-21314-696722	Certified	44510	15-1133.00	Amazon.com Services LLC	45411	WA	877
12	I-200-21314-696748	Certified	44510	15-1133.00	Amazon.com Services LLC	45422	WA	2193
13	I-200-21314-696780	Certified	44510	15-1133.00	Amazon.com Services LLC	45422	WA	2409
14	I-200-21314-696782	Certified	44510	15-1132.00	Amazon.com Services LLC	45422	WA	2295
15	I-200-21314-696937	Certified	44510	13-2011.00	Amazon.com Services LLC	45422	WA	877
16	I-200-21314-696997	Certified	44510	13-2011.00	Amazon.com Services LLC	45422	WA	754
17	I-200-21315-698187	Certified	44511	15-1132.00	Amazon.com Services LLC	45422	WA	2283
18	I-200-21315-698207	Certified	44511	15-1141.00	Amazon.com Services LLC	45422	WA	876
19	I-200-21315-698230	Certified	44511	15-1199.08	Amazon.com Services LLC	45422	WA	785

5. Filter out data that is unwanted
   I. Filter out employer_country and only show cases in the US
   II. Filter out states that are not part of the 50 states that are in the country, which are GU, MP, PR, PW, VI
   III. Filter Wage and only consider Yearly Salaries
6. Then we start creating the queries to normalize the data. The first query will be for application_status. We reference the "original"query and name it "application_status". We do this step for each query that has data we need to normalize (application_status, job_titles, states, industries and employers)

Application_Status

1. In application_status, remove columns and just keep CASE_STATUS and from it do the following:
   I. Transform > Trim the column CASE_STATUS
   II. Remove Duplicates
   III. Create Custom Index Column starting at "101"
   IV. Rename the columns

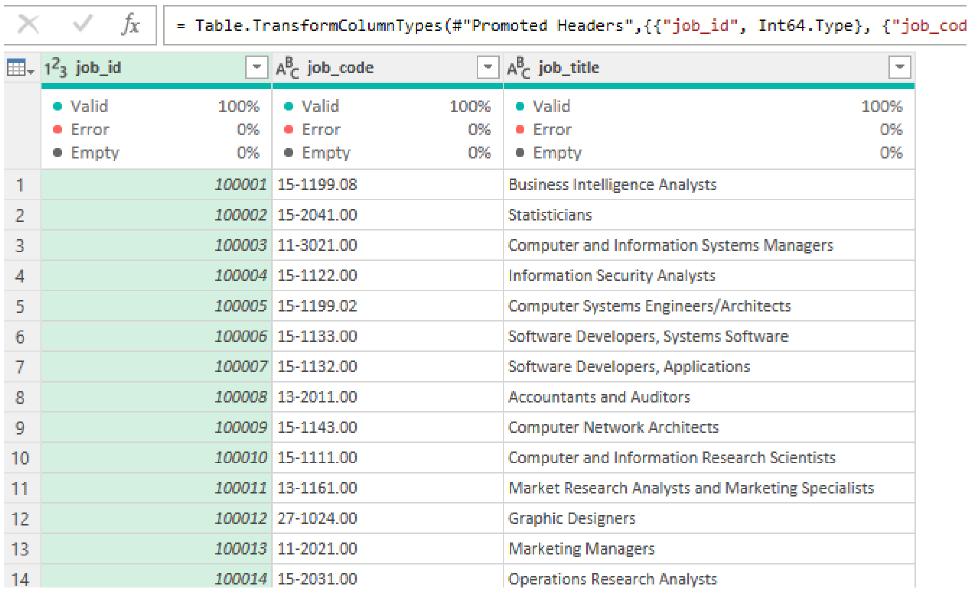


	status_id	case_status
	● Valid 100% ● Error 0% ● Empty 0%	● Valid 100% ● Error 0% ● Empty 0%
1	101	Certified
2	102	Withdrawn
3	103	Denied
4	104	Certified - Withdrawn

2. Back to case_application query, we must prepare the column to be merged with application_status.
   I. Transform > Trim CASE_STATUS
   II. Merge application_status (CASE_STATUS = case_status)
   III. Expand status_id
   IV. Remove other columns, just keep status_id

Job_Titles

1. In job_titles, remove the columns and keep SOC_CODE and SOC_TITLE and do the following:
   I. Transform > Trim the column SOC_TITLE
   II. Remove Duplicates
   III. Create Custom Index starting from "10000"
   IV. Rename the columns



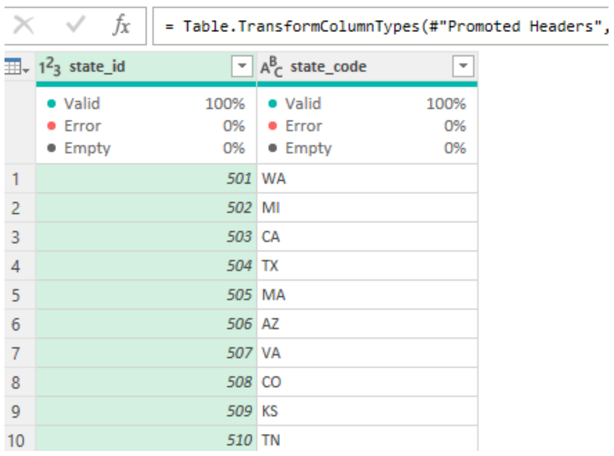
```
= Table.TransformColumnTypes(#"Promoted Headers",{{"job_id", Int64.Type}, {"job_cod
```

	job_id	job_code	job_title
	● Valid 100% ● Error 0% ● Empty 0%	● Valid 100% ● Error 0% ● Empty 0%	● Valid 100% ● Error 0% ● Empty 0%
1	100001	15-1199.08	Business Intelligence Analysts
2	100002	15-2041.00	Statisticians
3	100003	11-3021.00	Computer and Information Systems Managers
4	100004	15-1122.00	Information Security Analysts
5	100005	15-1199.02	Computer Systems Engineers/Architects
6	100006	15-1133.00	Software Developers, Systems Software
7	100007	15-1132.00	Software Developers, Applications
8	100008	13-2011.00	Accountants and Auditors
9	100009	15-1143.00	Computer Network Architects
10	100010	15-1111.00	Computer and Information Research Scientists
11	100011	13-1161.00	Market Research Analysts and Marketing Specialists
12	100012	27-1024.00	Graphic Designers
13	100013	11-2021.00	Marketing Managers
14	100014	15-2031.00	Operations Research Analysts

2. Back to case_application query, we must prepare the column to be merged with job_titles.
   I. Merge job_titles (SOC_Code = job_code)
   II. Expand job_id
   III. Remove other columns, just keep job_id

States

1. In state, remove all columns and just keep WORKSITE_STATE and do the following:
   I. Transform > Trim the only column
   II. Remove Duplicates
   III. Create Custom Index starting from "501"
   IV. Rename the columns



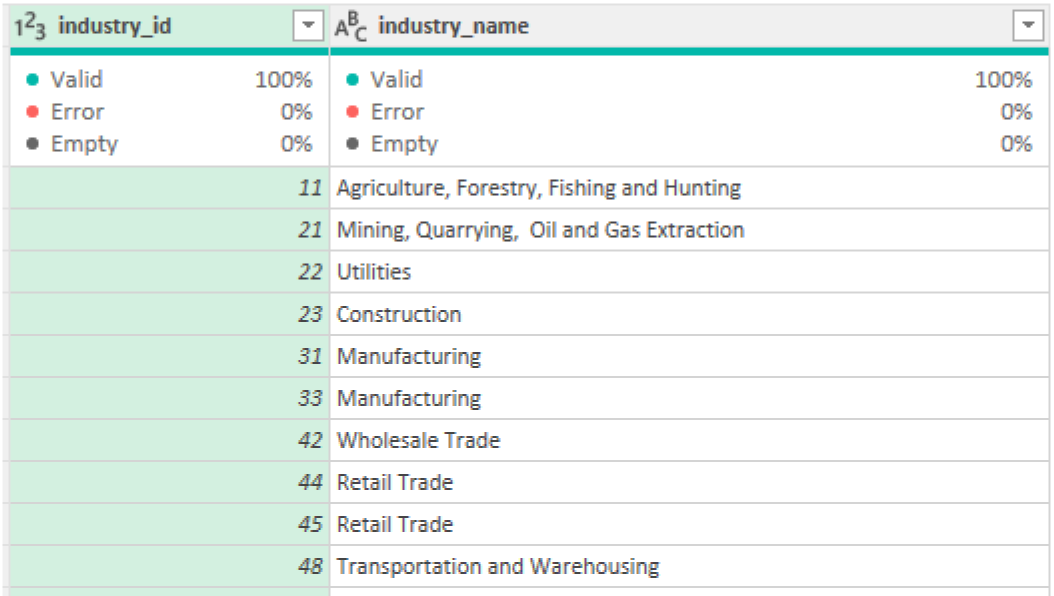
```
= Table.TransformColumnTypes(#"Promoted Headers",
```

	state_id	state_code
	● Valid 100% ● Error 0% ● Empty 0%	● Valid 100% ● Error 0% ● Empty 0%
1	501	WA
2	502	MI
3	503	CA
4	504	TX
5	505	MA
6	506	AZ
7	507	VA
8	508	CO
9	509	KS
10	510	TN

2. Back to case_application query, we must prepare the column to be merged with states.
   I. Transform > Trim WORKSITE_STATE
   II. Merge state (WORKSITE_STATE = state_code)
   III. Expand state_id
   IV. Remove other columns, just keep state_id

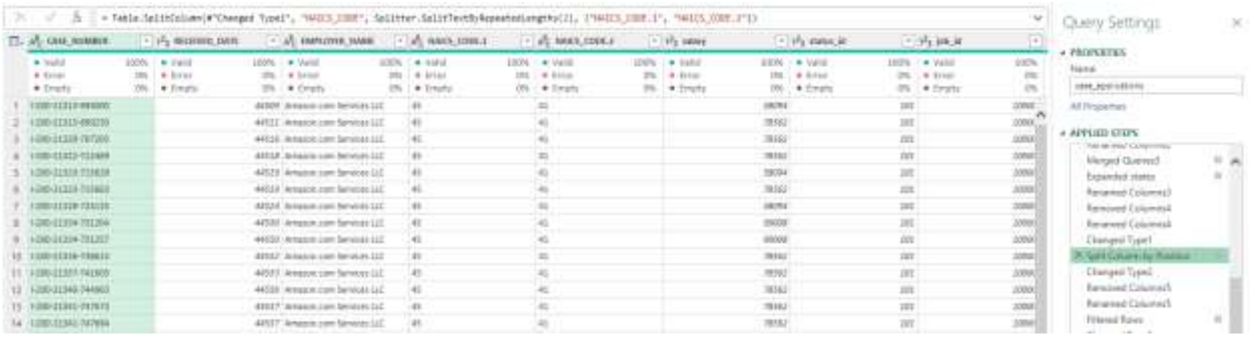
Industries

1. For Industry we created our own CSV table copied from the 2022 NAICS Website (https://www.census.gov/naics/?58967?yearbck=2022)
2. Upload the file as an additional query



industry_id	industry_name
● Valid 100% ● Error 0% ● Empty 0%	● Valid 100% ● Error 0% ● Empty 0%
11	Agriculture, Forestry, Fishing and Hunting
21	Mining, Quarrying, Oil and Gas Extraction
22	Utilities
23	Construction
31	Manufacturing
33	Manufacturing
42	Wholesale Trade
44	Retail Trade
45	Retail Trade
48	Transportation and Warehousing

3. Back to case_applications, we must prepare the column to merge with the official US NAICS codes
   I. Change NAICS column to Text
   II. Split Column by Number of Characters: 2



```
= Table.SplitColumn(#"Changed Type1", "NAICS_CODE", Splitter.SplitTextByRepeatedLengths(2), {"NAICS_CODE.1", "NAICS_CODE.2"})
```

Query Settings

PROPERTIES

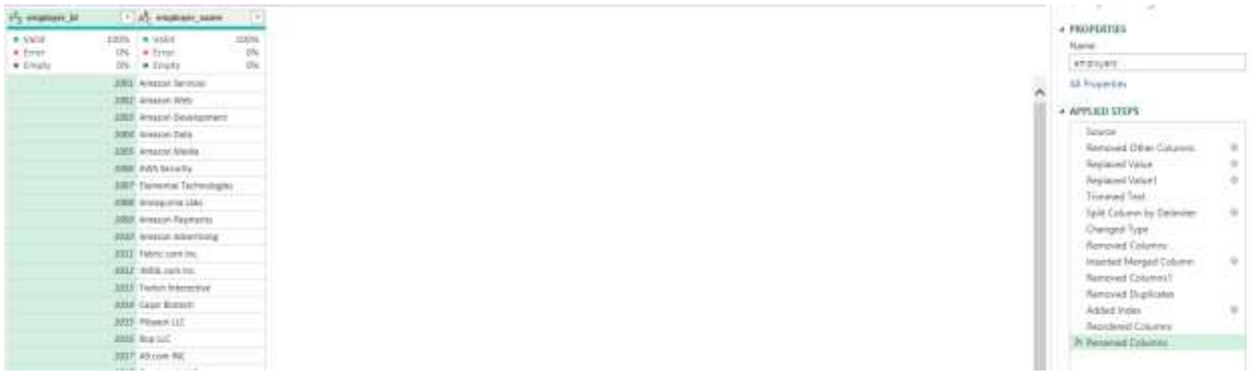
Name: case_applications

APPLIED STEPS: Merged Queries3, Expanded status, Renamed Columns3, Removed Columns4, Renamed Columns4, Changed Type1, Split Column by Position, Changed Type2, Removed Columns5, Renamed Columns5, Filtered Rows

III. Remove Columns NAICS_CODE2 and 3
IV. Filter out NAICS that are unknown (-2, 12 and 82)
V. Rename column industry_id

Employers

1. In employers, remove all columns and just keep EMPLOYER_NAME and do the following
   I. Replace values “,” for “”(empty)
   II. Replace values “Amazon.com” for “Amazon”
   III. Transform > Trim
   IV. Split Columns Delimeter by Space
   V. Remove columns, just keep the first two
   VI. Merge Columns 1 and 2 and name it employer_name
   VII. Remove Duplicates
   VIII. Add Index starting from 1001
   IX. Reorder Columns and rename them employer_id and employer_name



employer_id	employer_name
1001	Amazon Services
1002	Amazon Web
1003	Amazon Development
1004	Amazon Data
1005	Amazon Media
1006	AWS Security
1007	Elemental Technologies
1008	Amazon.com Labs
1009	Amazon Payments
1010	Amazon Advertising
1011	Fabric.com Inc
1012	IMDb.com Inc
1013	Twitch Interactive
1014	Cepir Biotech
1015	Pitech LLC
1016	Bop LLC
1017	A9.com INC

PROPERTIES

Name: employers

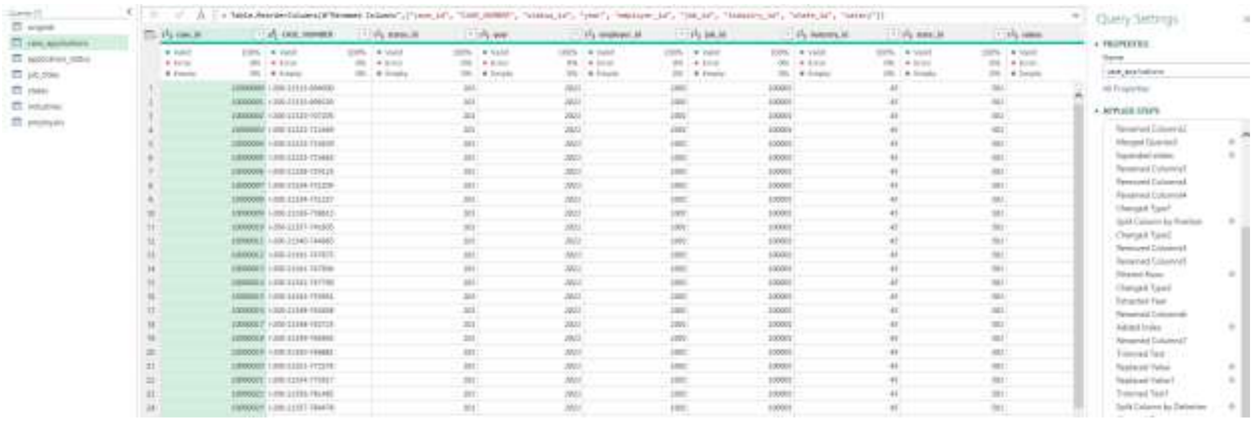
APPLIED STEPS: Source, Removed Other Columns, Replaced Value, Replaced Value1, Trimmed Text, Split Column by Delimiter, Changed Type, Removed Columns, Inserted Merged Column, Removed Columns1, Removed Duplicates, Added Index, Reordered Columns, Renamed Columns

2. Back to case_applications, we must prepare the column to be merge with employers. In the column EMPLOYER_NAME, do the following:
   I. Replace values “,” for “”(empty)
   II. Replace values “Amazon.com” for “Amazon”
   III. Transform > Trim
   IV. Split Columns Delimeter by Space
   V. Remove columns, just keep the first two
   VI. Merge Columns 1 and 2 and name it employer_name
   VII. Merge employers (employer_name = employer_name)
   VIII. Expand employer_id

IX. Remove other columns, just keep employer_id

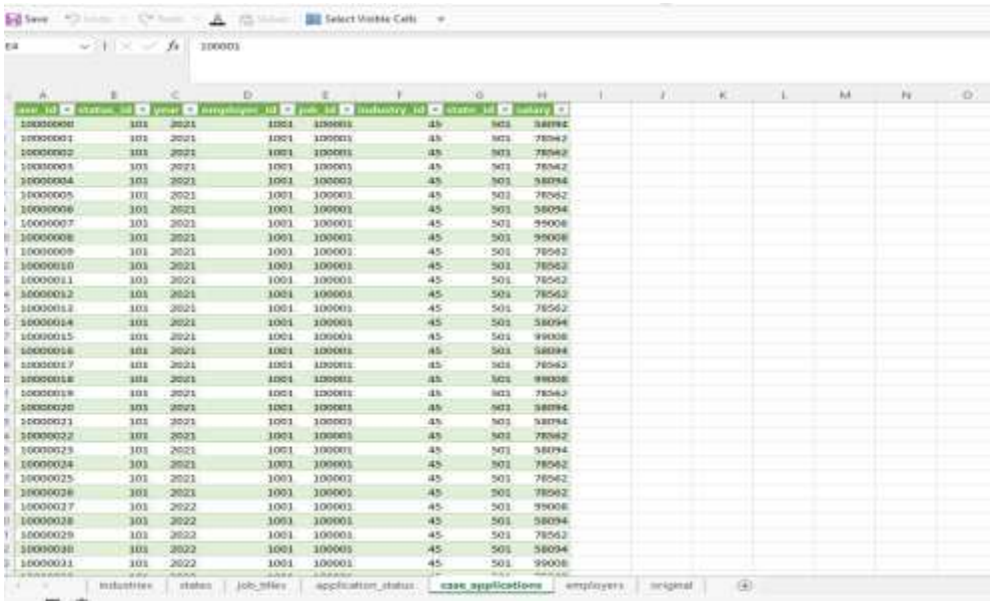
Final Steps in Power Query

1. Add Index in case_applications
2. Rename it case_id
3. Reorder columns and the case_applications query should look like this:



```
= Table.ReorderColumns(#"Renamed Columns",{"case_id", "CASE_NUMBER", "status_id", "year", "employer_id", "job_id", "industry_id", "state_id", "salary"})
```

4. Close and Load. Save the Excel File



E4 100001

case_id	status_id	year	employer_id	job_id	industry_id	state_id	salary
10000000	101	2021	1001	100001	45	501	58094
10000001	101	2021	1001	100001	45	501	78562
10000002	101	2021	1001	100001	45	501	78562
10000003	101	2021	1001	100001	45	501	78562
10000004	101	2021	1001	100001	45	501	58094
10000005	101	2021	1001	100001	45	501	78562
10000006	101	2021	1001	100001	45	501	58094
10000007	101	2021	1001	100001	45	501	99008
10000008	101	2021	1001	100001	45	501	99008
10000009	101	2021	1001	100001	45	501	78562
10000010	101	2021	1001	100001	45	501	78562
10000011	101	2021	1001	100001	45	501	78562
10000012	101	2021	1001	100001	45	501	78562
10000013	101	2021	1001	100001	45	501	78562
10000014	101	2021	1001	100001	45	501	58094
10000015	101	2021	1001	100001	45	501	99008
10000016	101	2021	1001	100001	45	501	58094
10000017	101	2021	1001	100001	45	501	78562
10000018	101	2021	1001	100001	45	501	99008
10000019	101	2021	1001	100001	45	501	78562
10000020	101	2021	1001	100001	45	501	58094
10000021	101	2021	1001	100001	45	501	58094
10000022	101	2021	1001	100001	45	501	78562
10000023	101	2021	1001	100001	45	501	58094
10000024	101	2021	1001	100001	45	501	78562
10000025	101	2021	1001	100001	45	501	78562
10000026	101	2021	1001	100001	45	501	78562
10000027	101	2022	1001	100001	45	501	99008
10000028	101	2022	1001	100001	45	501	58094
10000029	101	2022	1001	100001	45	501	78562
10000030	101	2022	1001	100001	45	501	58094
10000031	101	2022	1001	100001	45	501	99008

5. Open a New Excel File
6. Copy the data from Each tab
7. Paste Values in the 1. new Excel File
8. Save as CSV with the same name as the query title (ex: employers.csv)

PowerShell and Command Prompt

1. Using PowerShell transform each csv file using the following command

   import-csv 'C:\Users\aronq\Desktop\case_applications.csv' | export-csv 'C:\Users\aronq\Desktop\case_applications_bis.csv' -NoTypeInformation -Encoding UTF8

2. Open Command Prompt and go to the directory where all the bis.csv files are located (ex: cd Desktop)
3. Connect to the Server using the following command:

   mysql --local-infile=1 -show-warnings -h maksql.mysql.database.azure.com -u kpadhiyar -p

4. Upload Each file to the SQL Server using this command

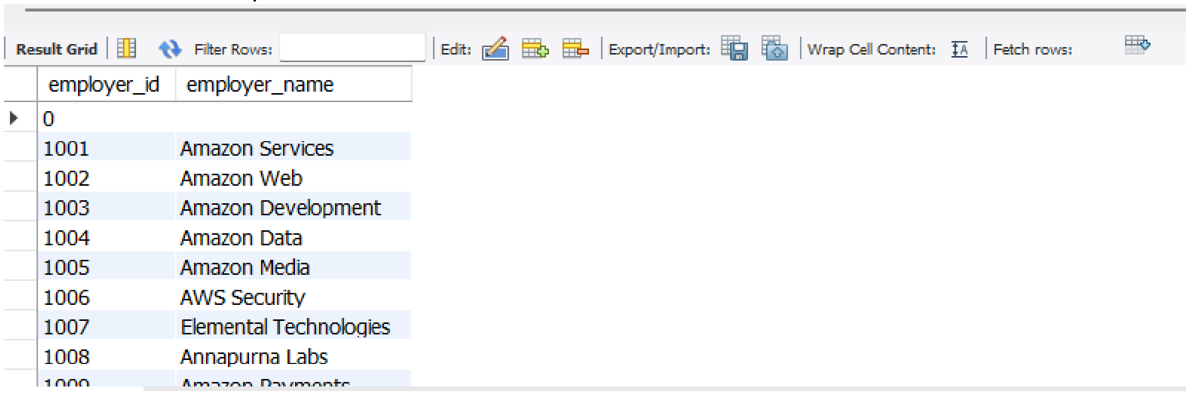
   USE h1b_visas;
   *LOAD DATA LOCAL INFILE **'employers_bis.csv'** INTO TABLE **`employers`** FIELDS TERMINATED BY ',' ENCLOSED BY '"' LINES TERMINATED BY '\r\n' IGNORE 1 LINES;*

   The words in bold can be edited to upload the rest of the files.

Checking if Data Upload Correctly

1. Go to MySQL and connect to the server that data was uploaded to.
2. Run the following Command
   USE h1b_visas;

   SELECT * **FROM** employers;
3. Check if the data is uploaded.



employer_id	employer_name
0	
1001	Amazon Services
1002	Amazon Web
1003	Amazon Development
1004	Amazon Data
1005	Amazon Media
1006	AWS Security
1007	Elemental Technologies
1008	Annapurna Labs
1009	Amazon Payments

Appendix

SQL Codes

```
# Top 3 occupations with highest sucessful visa sponsor over 5 years


USE h1b_visas;

SELECT ol.occupation_name, COUNT(ca.status_id) AS number_of_visa_success

FROM case_applications AS ca

INNER JOIN job_titles AS jt ON jt.job_id = ca.job_id

INNER JOIN occupation_list AS ol ON ol.occupation_id = jt.occupation_id

INNER JOIN application_status AS aps ON aps.status_id = ca.status_id

WHERE aps.status_id = 101

GROUP BY ol.occupation_name

ORDER BY number_of_visa_success DESC

LIMIT 3;


# Top 5 Employers in Massachusetts with highest visa sponsor rate


SELECT st.state_code, emp.employer_name, ind.industry_name, COUNT(ca.status_id) AS
number_of_visa_success

FROM case_applications AS ca

INNER JOIN states AS st ON st.state_id = ca.state_id

INNER JOIN employers AS emp ON emp.employer_id = ca.employer_id

INNER JOIN industries AS ind ON ind.industry_id = ca.industry_id

INNER JOIN application_status AS aps ON aps.status_id = ca.status_id

WHERE aps.status_id = 101

AND st.state_code = 'MA'

GROUP BY emp.employer_name, ind.industry_name, ca.state_id

ORDER BY number_of_visa_success DESC

LIMIT 5;
```

```
# Top 5 highest paying occupation on average over 5 years

SELECT ol.occupation_name, ROUND(AVG(ca.salary), 0) AS average_salary_per_year
FROM case_applications AS ca
INNER JOIN job_titles AS jt ON jt.job_id = ca.job_id
INNER JOIN occupation_list AS ol ON ol.occupation_id = jt.occupation_id
GROUP BY ol.occupation_name
ORDER BY average_salary_per_year DESC
LIMIT 5;

# Top 5 highest paying states with occupation name for the year 2022

SELECT st.state_code, ol.occupation_name, MAX(ca.salary) AS highest_salary_per_year
FROM case_applications AS ca
INNER JOIN job_titles AS jt ON jt.job_id = ca.job_id
INNER JOIN occupation_list AS ol ON ol.occupation_id = jt.occupation_id
INNER JOIN states AS st ON st.state_id = ca.state_id
WHERE `year` = 2022
GROUP BY st.state_code, ol.occupation_name
ORDER BY highest_salary_per_year DESC
LIMIT 5;

# Top 10 Employer with Highest visa success rate

SELECT emp.employer_name, ind.industry_name, COUNT(ca.status_id) AS number_of_visa_success
FROM case_applications AS ca
INNER JOIN employers AS emp ON emp.employer_id = ca.employer_id
INNER JOIN industries AS ind ON ind.industry_id = ca.industry_id
INNER JOIN application_status AS aps ON aps.status_id = ca.status_id
```

```sql
WHERE aps.status_id = 101

GROUP BY emp.employer_name, ind.industry_name

ORDER BY number_of_visa_success DESC

LIMIT 10;
```

Database Schema Code

```
-- MySQL Workbench Forward Engineering


SET @OLD_UNIQUE_CHECKS=@@UNIQUE_CHECKS, UNIQUE_CHECKS=0;

SET @OLD_FOREIGN_KEY_CHECKS=@@FOREIGN_KEY_CHECKS, FOREIGN_KEY_CHECKS=0;

SET @OLD_SQL_MODE=@@SQL_MODE,
SQL_MODE='ONLY_FULL_GROUP_BY,STRICT_TRANS_TABLES,NO_ZERO_IN_DATE,NO_ZERO_DATE,ERRO
R_FOR_DIVISION_BY_ZERO,NO_ENGINE_SUBSTITUTION';


-- -----------------------------------------------------

-- Schema mydb

-- -----------------------------------------------------

-- -----------------------------------------------------

-- Schema h1b_visas

-- -----------------------------------------------------



-- -----------------------------------------------------

-- Schema h1b_visas

-- -----------------------------------------------------

CREATE SCHEMA IF NOT EXISTS `h1b_visas` DEFAULT CHARACTER SET utf8 ;

USE `h1b_visas` ;


-- -----------------------------------------------------

-- Table `h1b_visas`.`application_status`

-- -----------------------------------------------------

CREATE TABLE IF NOT EXISTS `h1b_visas`.`application_status` (

  `status_id` INT NOT NULL,

  `case_status` VARCHAR(45) NULL DEFAULT NULL,

  PRIMARY KEY (`status_id`))
```

```
ENGINE = InnoDB
DEFAULT CHARACTER SET = utf8mb3;


-- -----------------------------------------------------
-- Table `h1b_visas`.`employers`
-- -----------------------------------------------------
CREATE TABLE IF NOT EXISTS `h1b_visas`.`employers` (
  `employer_id` INT NOT NULL,
  `employer_name` VARCHAR(200) NULL DEFAULT NULL,
  PRIMARY KEY (`employer_id`))
ENGINE = InnoDB
DEFAULT CHARACTER SET = utf8mb3;


-- -----------------------------------------------------
-- Table `h1b_visas`.`industries`
-- -----------------------------------------------------
CREATE TABLE IF NOT EXISTS `h1b_visas`.`industries` (
  `industry_id` INT NOT NULL,
  `industry_name` VARCHAR(200) NULL DEFAULT NULL,
  PRIMARY KEY (`industry_id`))
ENGINE = InnoDB
DEFAULT CHARACTER SET = utf8mb3;


-- -----------------------------------------------------
-- Table `h1b_visas`.`occupation_list`
-- -----------------------------------------------------
```

```
CREATE TABLE IF NOT EXISTS `h1b_visas`.`occupation_list` (
  `occupation_id` INT NOT NULL,
  `occupation_name` VARCHAR(100) NULL DEFAULT NULL,
  PRIMARY KEY (`occupation_id`))
ENGINE = InnoDB
DEFAULT CHARACTER SET = utf8mb3;


-- -----------------------------------------------------
-- Table `h1b_visas`.`job_titles`
-- -----------------------------------------------------
CREATE TABLE IF NOT EXISTS `h1b_visas`.`job_titles` (
  `job_id` INT NOT NULL,
  `occupation_id` INT NOT NULL,
  `job_title` VARCHAR(100) NULL DEFAULT NULL,
  PRIMARY KEY (`job_id`),
  INDEX `fk_job_titles_occupation_list1_idx` (`occupation_id` ASC) VISIBLE,
  CONSTRAINT `fk_job_titles_occupation_list1`
    FOREIGN KEY (`occupation_id`)
    REFERENCES `h1b_visas`.`occupation_list` (`occupation_id`))
ENGINE = InnoDB
DEFAULT CHARACTER SET = utf8mb3;


-- -----------------------------------------------------
-- Table `h1b_visas`.`states`
-- -----------------------------------------------------
CREATE TABLE IF NOT EXISTS `h1b_visas`.`states` (
  `state_id` INT NOT NULL,
  `state_code` VARCHAR(5) NULL DEFAULT NULL,
  PRIMARY KEY (`state_id`))
ENGINE = InnoDB
DEFAULT CHARACTER SET = utf8mb3;


-- -----------------------------------------------------
-- Table `h1b_visas`.`case_applications`
-- -----------------------------------------------------
CREATE TABLE IF NOT EXISTS `h1b_visas`.`case_applications` (
  `case_id` INT NOT NULL,
  `case_number` VARCHAR(100) NOT NULL,
  `status_id` INT NOT NULL,
  `year` YEAR NOT NULL,
  `employer_id` INT NOT NULL,
  `job_id` INT NOT NULL,
  `industry_id` INT NOT NULL,
  `state_id` INT NOT NULL,
  `salary` INT NOT NULL,
  PRIMARY KEY (`case_id`),
  INDEX `fk_case_applications_application_status1_idx` (`status_id` ASC) VISIBLE,
  INDEX `fk_case_applications_industries1_idx` (`industry_id` ASC) VISIBLE,
  INDEX `fk_case_applications_employers1_idx` (`employer_id` ASC) VISIBLE,
  INDEX `fk_case_applications_states1_idx` (`state_id` ASC) VISIBLE,
  INDEX `fk_case_applications_job_titles1_idx` (`job_id` ASC) VISIBLE,
  CONSTRAINT `fk_case_applications_application_status1`
    FOREIGN KEY (`status_id`)
    REFERENCES `h1b_visas`.`application_status` (`status_id`),
  CONSTRAINT `fk_case_applications_employers1`
```

```
    FOREIGN KEY (`employer_id`)
    REFERENCES `h1b_visas`.`employers` (`employer_id`),
  CONSTRAINT `fk_case_applications_industries1`
    FOREIGN KEY (`industry_id`)
    REFERENCES `h1b_visas`.`industries` (`industry_id`),
  CONSTRAINT `fk_case_applications_job_titles1`
    FOREIGN KEY (`job_id`)
    REFERENCES `h1b_visas`.`job_titles` (`job_id`),
  CONSTRAINT `fk_case_applications_states1`
    FOREIGN KEY (`state_id`)
    REFERENCES `h1b_visas`.`states` (`state_id`))
ENGINE = InnoDB
DEFAULT CHARACTER SET = utf8mb3;



SET SQL_MODE=@OLD_SQL_MODE;
SET FOREIGN_KEY_CHECKS=@OLD_FOREIGN_KEY_CHECKS;
SET UNIQUE_CHECKS=@OLD_UNIQUE_CHECKS;
```