```
In [1]: ! pip install -U numpy
        ! pip install missingno
```

```
Collecting numpy
  Downloading https://files.pythonhosted.org/packages/45/b2/6c7545bb7a38754d63048c7696804a0d947328125d81bf12beaa692c3ae3/numpy-1.19.5-cp36-cp36m-manylinux1_x86_64.whl (13.4MB)
    100% |████████████████████████████████| 13.4MB 1.4MB/s eta 0:00:01   24% |███████▉                        | 3.2MB 33.5MB/s eta 0:00:01    36% |███████████▊                    | 4.8MB 32.1MB/s eta 0:00:01    47% |███████████████▏                | 6.4MB 31.2MB/s eta 0:00:01    81% |██████████████████████████      | 10.9MB 32.1MB/s eta 0:00:01
tensorflow 1.3.0 requires tensorflow-tensorboard<0.2.0,>=0.1.0, which is not installed.
Installing collected packages: numpy
  Found existing installation: numpy 1.12.1
    Uninstalling numpy-1.12.1:
      Successfully uninstalled numpy-1.12.1
Successfully installed numpy-1.19.5
Collecting missingno
  Downloading https://files.pythonhosted.org/packages/87/22/cd5cf999af21c2f97486622c551ac3d07361ced8125121e907f588ff5f24/missingno-0.5.2-py3-none-any.whl
Requirement already satisfied: numpy in /opt/conda/lib/python3.6/site-packages (from missingno) (1.19.5)
Requirement already satisfied: scipy in /opt/conda/lib/python3.6/site-packages (from missingno) (1.2.1)
Requirement already satisfied: seaborn in /opt/conda/lib/python3.6/site-packages (from missingno) (0.8.1)
Requirement already satisfied: matplotlib in /opt/conda/lib/python3.6/site-packages (from missingno) (2.1.0)
Requirement already satisfied: pandas in /opt/conda/lib/python3.6/site-packages (from seaborn->missingno) (0.23.3)
Requirement already satisfied: six>=1.10 in /opt/conda/lib/python3.6/site-packages (from matplotlib->missingno) (1.11.0)
Requirement already satisfied: python-dateutil>=2.0 in /opt/conda/lib/python3.6/site-packages (from matplotlib->missingno) (2.6.1)
Requirement already satisfied: pytz in /opt/conda/lib/python3.6/site-packages (from matplotlib->missingno) (2017.3)
Requirement already satisfied: cycler>=0.10 in /opt/conda/lib/python3.6/site-packages/cycler-0.10.0-py3.6.egg (from matplotlib->missingno) (0.10.0)
Requirement already satisfied: pyparsing!=2.0.4,!=2.1.2,!=2.1.6,>=2.0.1 in /opt/conda/lib/python3.6/site-packages (from matplotlib->missingno) (2.2.0)
Installing collected packages: missingno
Successfully installed missingno-0.5.2
```

```
In [2]: import pandas as pd
import matplotlib.pyplot as plt
import os
import configparser
import datetime as dt
from pyspark.sql.functions import isnan, when, count, col, udf, dayofmonth, dayofweek, month, year, weekofyear, avg, monotonically_increasing_id
from pyspark.sql.types import *
import requests
requests.packages.urllib3.disable_warnings()
from pyspark.sql.functions import year, month, dayofmonth, weekofyear, date_format
from pyspark.sql import SparkSession, SQLContext, GroupedData, HiveContext
from pyspark.sql.functions import *
from pyspark.sql.functions import date_add as d_add
from pyspark.sql.types import DoubleType, StringType, IntegerType, FloatType
from pyspark.sql import functions as F
from pyspark.sql.functions import lit
from pyspark.sql import Row
import datetime, time
import seaborn as sns
import numpy as np
import tools as tools
import tables_func as ct
import matplotlib.pyplot as plt
%matplotlib inline
```

Scope of the Project:

- In this project I will gather the data from two sources. I will load this data into staging dataframes. I will clean the raw data, write it to parquet files and perform an ETL process using a Spark cluster. Then I will write the data into Fact & Dimension tables to form a star schema. The star schema can then be used by the relevant parties to perform data analytics, correlation and ad-hoc reporting in an effective and efficient manner. ### Data Descriptions:
- Edu_ministry is Sample data of saudi scholarships students records from the Ministry of Education. This data source will serve as the Fact table in the schema. This data comes from https://od.data.gov.sa/Data/ar/group/education_and_training?page=2 :'البوابة الوطنية للبيانات المفتوحة'.
- Countries of the World :This dataset contains Country names linked to region, population, area size, GDP, mortality and more.

  This data comes from Kaggle: https://www.kaggle.com/datasets/fernandol/countries-of-the-world

```
In [3]:     spark = SparkSession\
        .builder \
        .config("spark.jars.packages", "org.apache.hadoop:hadoop-aws:2.7.0") \
        .getOrCreate()
```

```
In [5]: #import shutil

#shutil.rmtree('../Project/parquet_tables/countries')
```

Spark configuration parameters

- ('spark.executor.id', 'driver'),
- ('spark.app.name', 'pyspark-shell'),
- ('spark.driver.port', '33753'),
- ('spark.jars.packages', 'org.apache.hadoop:hadoop-aws:2.7.0'),
- ('spark.rdd.compress', 'True'),
- ('spark.serializer.objectStreamReset', '100'),
- ('spark.master', 'local[*]'),
- ('spark.driver.host', 'ad2cb8c219e1'),
- ('spark.submit.deployMode', 'client'),
- ('spark.jars',

```
In [6]: fname = '../Project/Sources/Edu_minisrty.csv'
edu_df = pd.read_csv(fname)
```

```
In [7]: edu_df.head()
```

Out[7]:

	السنة الدراسية	القطاع الدولي	فئة الإبتعاث	المرحلة الدراسية	المجال الواسع	الجنس	العدد
0	2015.0	استراليا	طالب مبتعث	أخرى	الأعمال، الإدارة والقانون	أنثى	11.0
1	2015.0	استراليا	طالب مبتعث	أخرى	الأعمال، الإدارة والقانون	ذكر	6.0
2	2015.0	استراليا	طالب مبتعث	أخرى	التعليم	أنثى	4.0
3	2015.0	استراليا	طالب مبتعث	أخرى	الصحة والرفاه	أنثى	3.0
4	2015.0	استراليا	طالب مبتعث	أخرى	العلوم الطبيعية، والرياضايات والإحصاء	ذكر	1.0

```
In [8]: fname = '../Project/Sources/countries of the world.csv'
country_df = pd.read_csv(fname)
```

```
In [9]: country_df.head()
```

Out[9]:

	Country	Region	Population	Area (sq. mi.)	Pop. Density (per sq. mi.)	Coastline (coast/area ratio)	Net migration	Infant mortality (per 1000 births)	GDP ($ per capita)	Literacy (%)	Phones (per 1000)	Arable (%)	Crops (%)	Other (%)	Climate	Birthrate
0	Afghanistan	ASIA (EX. NEAR EAST)	31056997	647500	48,0	0,00	23,06	163,07	700.0	36,0	3,2	12,13	0,22	87,65	1	46,6
1	Albania	EASTERN EUROPE	3581655	28748	124,6	1,26	-4,93	21,52	4500.0	86,5	71,2	21,09	4,42	74,49	3	15,11
2	Algeria	NORTHERN AFRICA	32930091	2381740	13,8	0,04	-0,39	31	6000.0	70,0	78,1	3,22	0,25	96,53	1	17,14
3	American Samoa	OCEANIA	57794	199	290,4	58,29	-20,71	9,27	8000.0	97,0	259,5	10	15	75	2	22,46
4	Andorra	WESTERN EUROPE	71201	468	152,1	0,00	6,6	4,05	19000.0	100,0	497,2	2,22	0	97,78	3	8,71

Exploration and Assessing the Data

Data Cleaning:

- Drop columns containing over 90% missing values
- Drop duplicate values

```
In [10]: # checking missing values
edu_df.isnull().sum()
```

```
Out[10]: 0        السنة الدراسية
0        القطاع الدولي
0        فئة الإبتعاث
0        المرحلة الدراسية
0        المجال الواسع
0        الجنس
0        العدد
dtype: int64
```

```
In [11]: # Drop columns with over 90% missing values
clean_edu = tools.eliminate_missing_data(edu_df)

Dropping missing data...
Cleaning complete!
```

```
In [12]: clean_edu = tools.drop_duplicate_rows(clean_edu)

Dropping duplicate rows...
0 rows dropped.
```

```
In [13]: # Drop columns with over 90% missing values
clean_country = tools.eliminate_missing_data(country_df)

Dropping missing data...
Cleaning complete!
```

```
In [14]: clean_country = tools.drop_duplicate_rows(clean_country)

Dropping duplicate rows...
0 rows dropped.
```

Now I'll translate the dataset to english language

```
In [15]: clean_edu = ct.translate_en(clean_edu)
clean_edu.head()
```

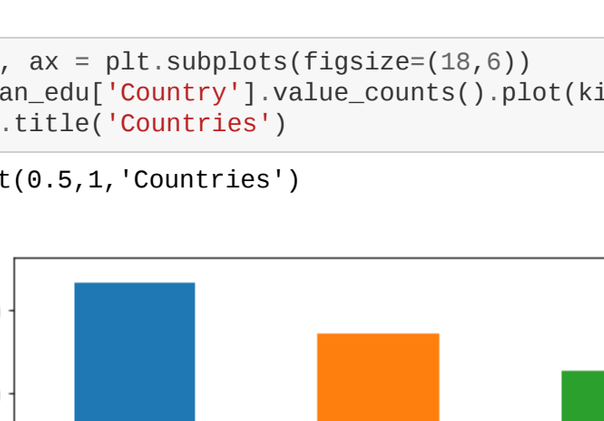
Out[15]:

	Academic_year	Country	Scholarship_type	Educational_level	Specialization	Sex	Count
0	2015.0	Australia	Scholarship student	Other	Business, Management and Law	2	11.0
1	2015.0	Australia	Scholarship student	Other	Business, Management and Law	1	6.0
2	2015.0	Australia	Scholarship student	Other	Education	2	4.0
3	2015.0	Australia	Scholarship student	Other	Health and well-being	2	3.0
4	2015.0	Australia	Scholarship student	Other	Natural sciences, mathematics and statistics	1	1.0

```
In [16]: clean_edu['Country'].count()
Out[16]: 3148
```

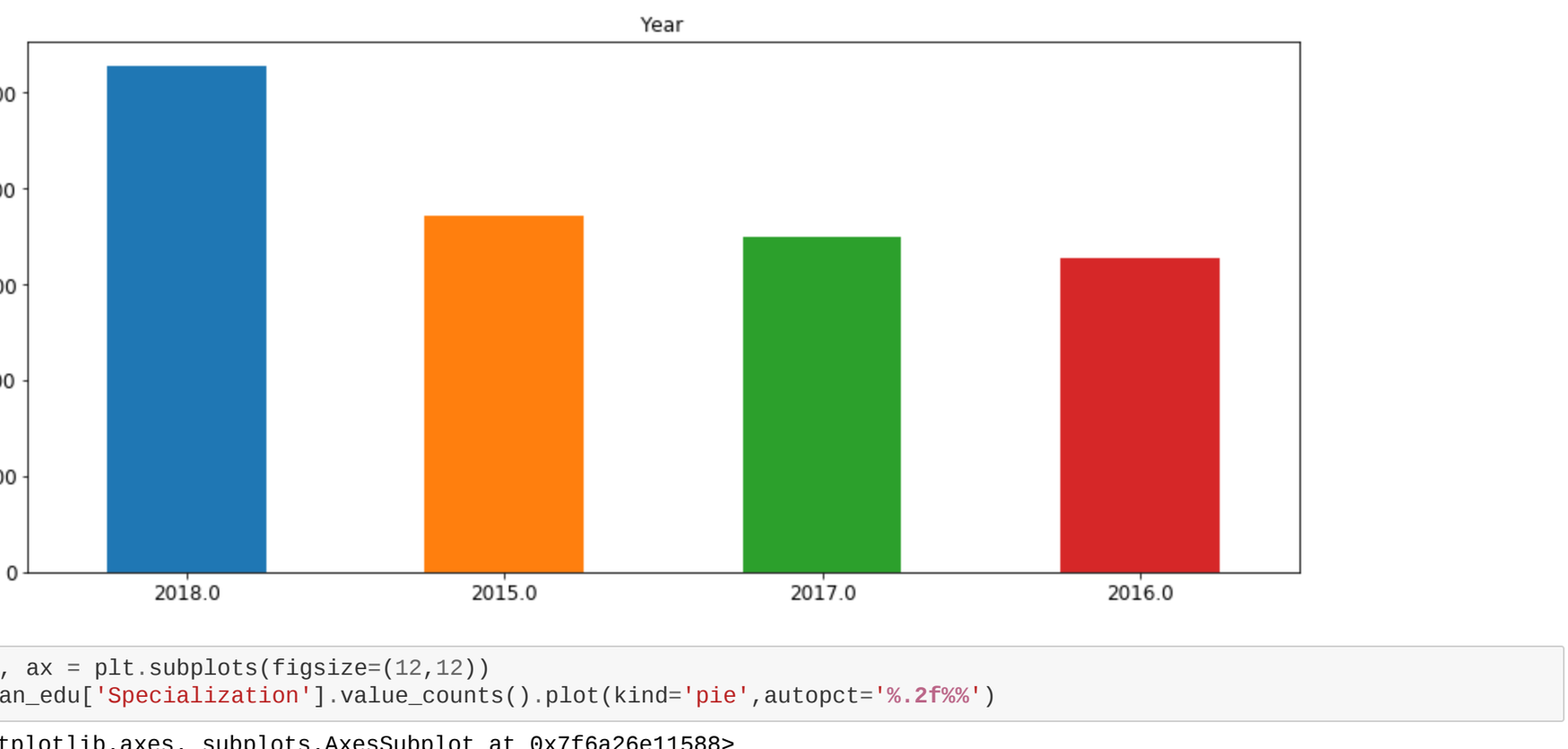
Males represent No.1 and females No.2

```
In [15]: clean_edu['Sex'].value_counts().plot(kind='pie',autopct='%.2f%%')
Out[15]: <matplotlib.axes._subplots.AxesSubplot at 0x7f501e43fda0>
```

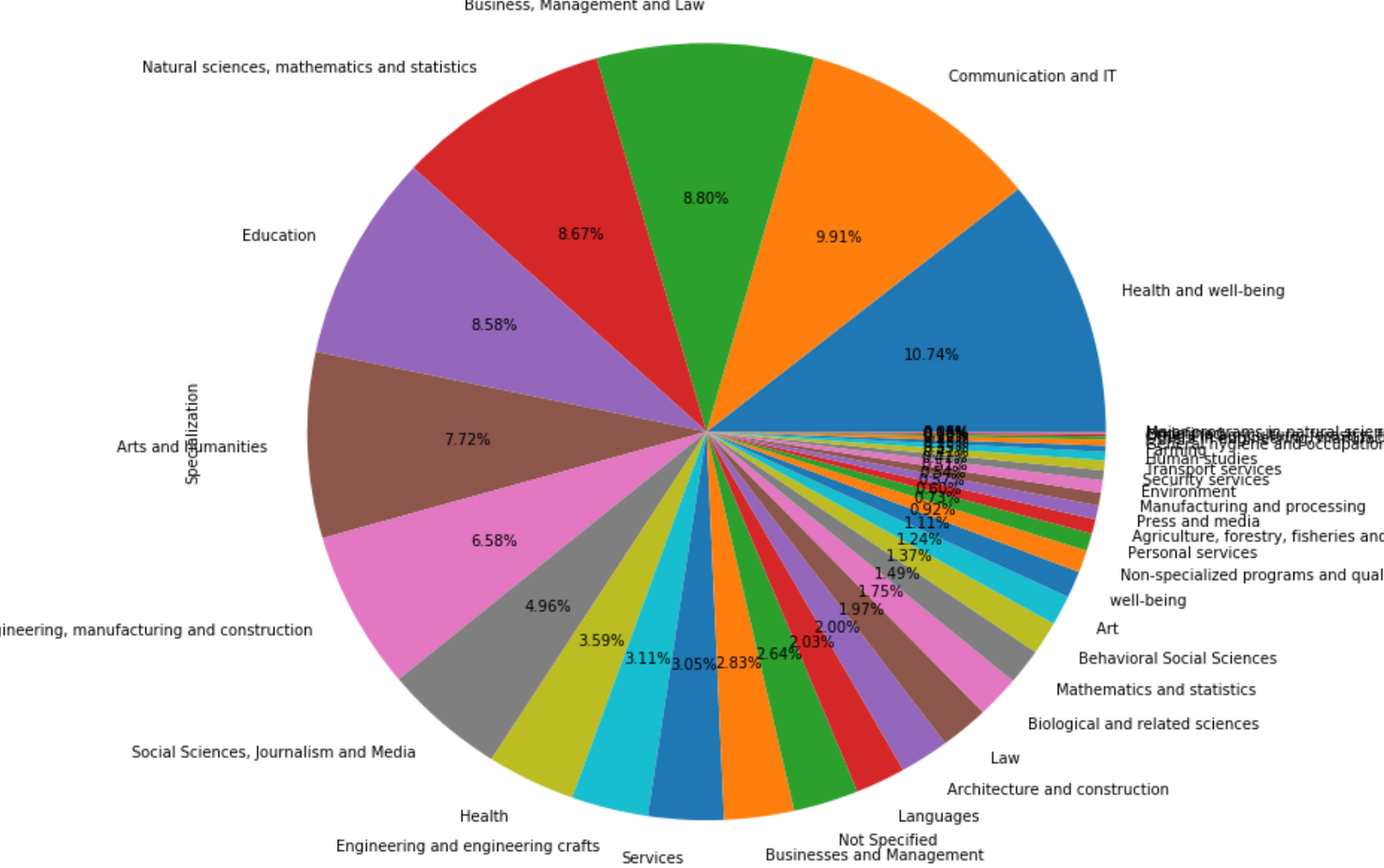



```
In [17]: fig, ax = plt.subplots(figsize=(16,4))
clean_edu['Scholarship_type'].value_counts().plot(kind='bar', fontsize =12, rot = 0)
plt.title('Scholarship Types')
Out[17]: Text(0.5,1,'Scholarship Types')
```

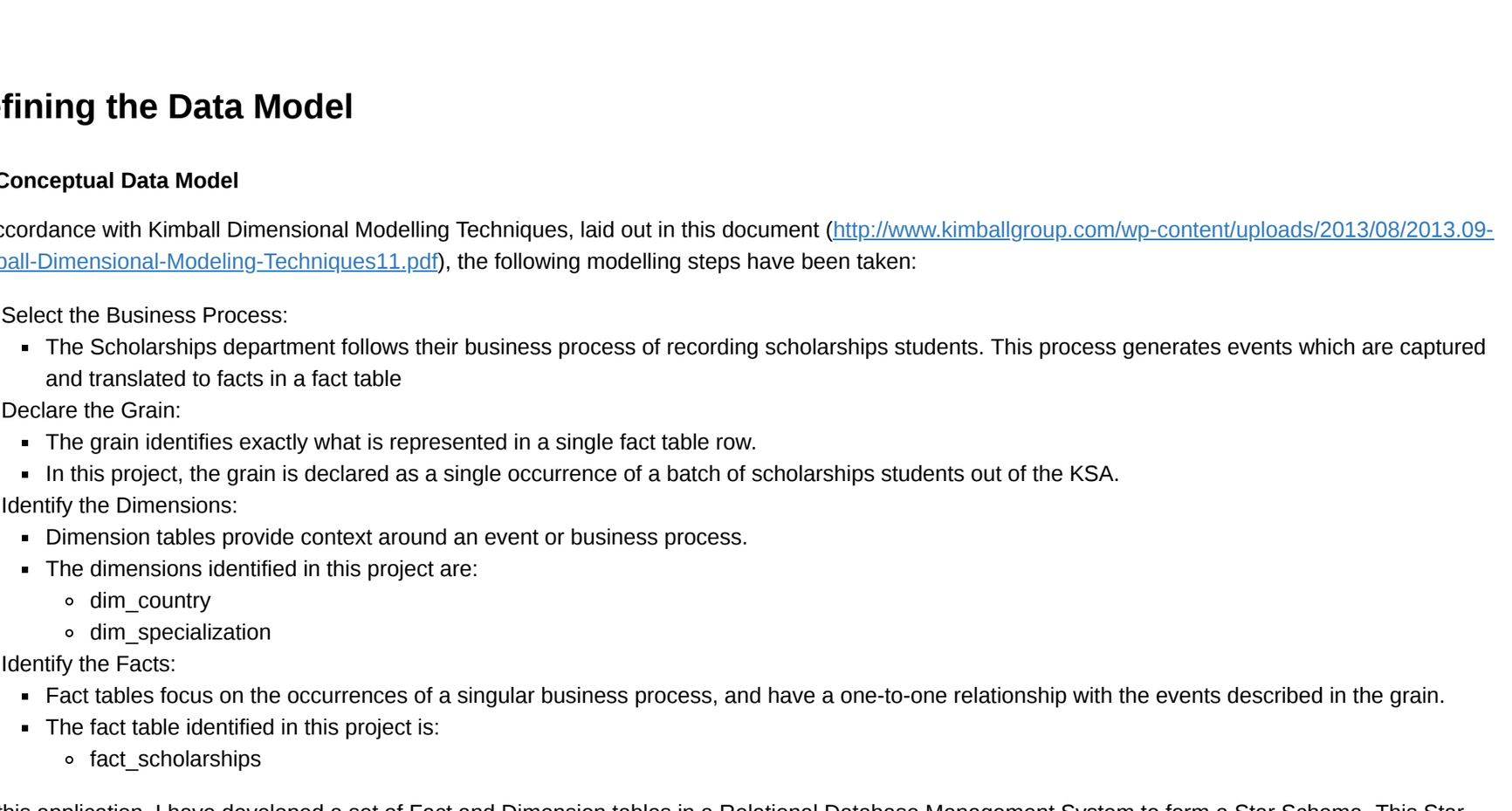

```
In [18]: fig, ax = plt.subplots(figsize=(18,6))
clean_edu['Country'].value_counts().plot(kind='bar', fontsize =12, rot = 0)
plt.title('Countries')
Out[18]: Text(0.5,1,'Countries')
```




```
In [19]: fig, ax = plt.subplots(figsize=(14,6))
clean_edu['Academic_year'].value_counts().plot(kind='bar', fontsize =12, rot = 0)
plt.title('Year')
Out[19]: Text(0.5,1,'Year')
```




```
In [20]: fig, ax = plt.subplots(figsize=(12,12))
clean_edu['Specialization'].value_counts().plot(kind='pie',autopct='%.2f%%')
Out[20]: <matplotlib.axes._subplots.AxesSubplot at 0x7f6a26e11588>
```



Defining the Data Model

3.1 Conceptual Data Model

In accordance with Kimball Dimensional Modelling Techniques, laid out in this document (http://www.kimballgroup.com/wp-content/uploads/2013/08/2013.09-Kimball-Dimensional-Modeling-Techniques11.pdf), the following modelling steps have been taken:

- Select the Business Process:
  - The Scholarships department follows their business process of recording scholarships students. This process generates events which are captured and translated to facts in a fact table
- Declare the Grain:
  - The grain identifies exactly what is represented in a single fact table row.
  - In this project, the grain is declared as a single occurrence of a batch of scholarships students out of the KSA.
- Identify the Dimensions:
  - Dimension tables provide context around an event or business process.
  - The dimensions identified in this project are:
    - dim_country
    - dim_specialization
- Identify the Facts:
  - Fact tables focus on the occurrences of a singular business process, and have a one-to-one relationship with the events described in the grain.
  - The fact table identified in this project is:
    - fact_scholarships

For this application, I have developed a set of Fact and Dimension tables in a Relational Database Management System to form a Star Schema. This Star Schema can be used by Data Analysts and other relevant business professionals to gain deeper insight into various scholarships figures, trends and statistics recorded historically.

3.2 Mapping Out Data Pipelines

List the steps necessary to pipeline the data into the chosen data model:

- 1 Load the data into staging tables
- 2 Create Dimension tables
- 3 Create Fact table
- 4 Write data into parquet files
- 5 Perform data quality checks

Running Pipelines to Model the Data

4.1 Create the data model

Build the data pipelines to create the data model.

```
In [17]: output_path = "parquet_tables/"
```

```
In [18]: clean_edu.head()
```

Out[18]:

	Academic_year	Country	Scholarship_type	Educational_level	Specialization	Sex	Count
0	2015.0	Australia	Scholarship student	Other	Business, Management and Law	2	11.0
1	2015.0	Australia	Scholarship student	Other	Business, Management and Law	1	6.0
2	2015.0	Australia	Scholarship student	Other	Education	2	4.0
3	2015.0	Australia	Scholarship student	Other	Health and well-being	2	3.0
4	2015.0	Australia	Scholarship student	Other	Natural sciences, mathematics and statistics	1	1.0

```
In [19]: # creating schema
scholarships_schema = StructType([StructField("Academic_year", StringType(), True)\
                        ,StructField("Country", StringType(), True)\
                        ,StructField("Scholarship_type", StringType(), True)\
                        ,StructField("Educational_level", StringType(), True)\
                        ,StructField("Specialization", StringType(), True)\
                        ,StructField("Sex", StringType(), True)\
                        ,StructField("Count", FloatType(), True)])

scholarships_spark = spark.createDataFrame(clean_edu, schema=scholarships_schema)

scholarships_spark.toPandas().head()
```

Out[19]:

	Academic_year	Country	Scholarship_type	Educational_level	Specialization	Sex	Count
0	2015.0	Australia	Scholarship student	Other	Business, Management and Law	2	11.0
1	2015.0	Australia	Scholarship student	Other	Business, Management and Law	1	6.0
2	2015.0	Australia	Scholarship student	Other	Education	2	4.0
3	2015.0	Australia	Scholarship student	Other	Health and well-being	2	3.0
4	2015.0	Australia	Scholarship student	Other	Natural sciences, mathematics and statistics	1	1.0

```
In [31]: clean_country.head()
```

Out[31]:

	Country	Region	Population	Area (sq. mi.)	Pop. Density (per sq. mi.)	Coastline (coast/area ratio)	Net migration	Infant mortality (per 1000 births)	GDP ($ per capita)	Literacy (%)	Phones (per 1000)	Arable (%)	Crops (%)	Other (%)	Climate	Birthrate
0	Afghanistan	ASIA (EX. NEAR EAST)	31056997	647500	48,0	0,00	23,06	163,07	700.0	36,0	3,2	12,13	0,22	87,65	1	46,6
1	Albania	EASTERN EUROPE	3581655	28748	124,6	1,26	-4,93	21,52	4500.0	86,5	71,2	21,09	4,42	74,49	3	15,11
2	Algeria	NORTHERN AFRICA	32930091	2381740	13,8	0,04	-0,39	31	6000.0	70,0	78,1	3,22	0,25	96,53	1	17,14
3	American Samoa	OCEANIA	57794	199	290,4	58,29	-20,71	9,27	8000.0	97,0	259,5	10	15	75	2	22,46
4	Andorra	WESTERN EUROPE	71201	468	152,1	0,00	6,6	4,05	19000.0	100,0	497,2	2,22	0	97,78	3	8,71

```
In [21]: #"Country","Region","Population","Net migration","Coastline (coast/area ratio)","GDP ($ per capita)
country_schema = StructType([StructField("Country", StringType(), True)\
                          ,StructField("Region", StringType(), True)\
                          ,StructField("Population", IntegerType(), True)\
                          ,StructField("Area (sq. mi.)", IntegerType(), True)\
                          ,StructField("Pop. Density (per sq. mi.)", StringType(), True)\
                          ,StructField("Coastline (coast/area ratio)", StringType(), True)\
                          ,StructField("Net migration)", StringType(), True)\
                          ,StructField("Infant mortality (per 1000 births)", StringType(), True)\
                          ,StructField("GDP ($ per capita)", FloatType(), True)
                         ,StructField("Literacy (%)", StringType(), True)\
                         ,StructField("Phones (per 1000)", StringType(), True)\
                         ,StructField("Arable (%)", StringType(), True)\
                         ,StructField("Crops (%)", StringType(), True)\
                         ,StructField("Other (%)", StringType(), True)\
                         ,StructField("Climate", StringType(), True)\
                         ,StructField("Birthrate", StringType(), True)\
                         ,StructField("Deathrate", StringType(), True)\
                         ,StructField("Agriculture", StringType(), True)\
                         ,StructField("ndustry", StringType(), True)\
                         ,StructField("Service", StringType(), True)])

country_spark = spark.createDataFrame(clean_country, schema=country_schema)

country_spark.toPandas().head()
```

Out[21]:

	Country	Region	Population	Area (sq. mi.)	Pop. Density (per sq. mi.)	Coastline (coast/area ratio	Net migration)	Infant mortality (per 1000 births)	GDP ($ per capita	Literacy (%)	Phones (per 1000)	Arable (%)	Crops (%)	Other (%)	Climate	Birthrate
0	Afghanistan	ASIA (EX. NEAR EAST)	31056997	647500	48,0	0,00	23,06	163,07	700.0	36,0	3,2	12,13	0,22	87,65	1	46,6
1	Albania	EASTERN EUROPE	3581655	28748	124,6	1,26	-4,93	21,52	4500.0	86,5	71,2	21,09	4,42	74,49	3	15,11
2	Algeria	NORTHERN AFRICA	32930091	2381740	13,8	0,04	-0,39	31	6000.0	70,0	78,1	3,22	0,25	96,53	1	17,14
3	American Samoa	OCEANIA	57794	199	290,4	58,29	-20,71	9,27	8000.0	97,0	259,5	10	15	75	2	22,46
4	Andorra	WESTERN EUROPE	71201	468	152,1	0,00	6,6	4,05	19000.0	100,0	497,2	2,22	0	97,78	3	8,71

```
In [22]: # inserting missed rows
added_row = [['Arab Countries', 'MIDLE EAST',0,0,'','','','',0.0,'','','','','','','','','','',''],]\
        ,['Asian Countries', 'ASIA',0,0,'','','','',0.0,'','','','','','','','','','',''],]\
        ,['European Countries', 'EUROPE',0,0,'','','','',0.0,'','','','','','','','','','','',]]

# Creating the DataFrame
newRow = spark.createDataFrame(added_row,schema=None)
country_spark= country_spark.union(newRow)
```

1. Create dim_Country table

```
In [23]: ct.create_country_dim(country_spark, output_path)

Writing table countries to parquet_tables/countries
Write complete!

Out[23]: DataFrame[country_id: int, country_name: string, Region: string, Population: bigint, Coastline_area_ratio: string, Net_migration: string, GDP_$_per_capital: double]
```

```
In [24]: country = spark.read.parquet("parquet_tables/countries")

country.toPandas().head()
```

Out[24]:

	country_id	country_name	Region	Population	Coastline_area_ratio	Net_migration	GDP_$_per_capital
0	0	Afghanistan	ASIA (EX. NEAR EAST)	31056997	0,00	23,06	700.0
1	1	Albania	EASTERN EUROPE	3581655	1,26	-4,93	4500.0
2	2	Algeria	NORTHERN AFRICA	32930091	0,04	-0,39	6000.0
3	3	American Samoa	OCEANIA	57794	58,29	-20,71	8000.0
4	4	Andorra	WESTERN EUROPE	71201	0,00	6,6	19000.0

2. Create dim_Specialization table

```
In [25]: ct.create_specialization_dim(scholarships_spark, output_path)

Writing table specializations to parquet_tables/specializations
Write complete!

Out[25]: DataFrame[spec_id: int, spec_desc: string]
```

```
In [41]: specialization = spark.read.parquet("parquet_tables/specializations")

specialization.toPandas().head()
```

Out[41]:

	spec_id	spec_desc
0	0	Services
1	1	Engineering, manufacturing and construction
2	2	Education
3	3	Farming
4	4	Others in engineering, manufacturing and const...

3. Create fact_Scholarships table

```
In [27]: scholarships = ct.create_scholarships_fact(scholarships_spark, output_path, spark)

Writing table scholarships to parquet_tables/scholarships
Write complete!
```

```
In [28]: scholarships = spark.read.parquet("parquet_tables/scholarships")

scholarships.toPandas().head()
```

Out[28]:

	scholarships_id	spec_id	country_id	Academic_year	Educational_level	Sex	Count
0	0	0	228	2015.0	Other	1	30.0
1	1	33	228	2015.0	Other	2	1.0
2	2	33	228	2015.0	Other	1	1.0
3	3	1	228	2015.0	Other	1	2.0
4	4	16	228	2015.0	Fellowship	1	2.0

```
In [29]: scholarships.count()
Out[29]: 1738
```

Data Quality Checks

```
In [5]: tables = {
    "specialization": specialization,
    "country": country,
    "scholarships": scholarships
}

for table_name, table in tables.items():
    tools.perform_quality_check(table, table_name)
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
<ipython-input-5-56fc45cfc6ef> in <module>()
      1 tables = {
----> 2     "specialization": specialization,
      3     "country": country,
      4     "scholarships": scholarships
      5 }

NameError: name 'specialization' is not defined
```

```
In [6]: countriesTable = spark.read.parquet("parquet_tables/countries")
specializationTable = spark.read.parquet("parquet_tables/specializations")
scholarshipsTable = spark.read.parquet('parquet_tables/scholarships')
```

```
In [30]: #Females
scholarshipsTable.filter(scholarshipsTable.Sex == '2') \
                .select('scholarships_id', 'Sex') \
                .dropDuplicates() \
                .count()
Out[30]: 828
```

```
In [31]: # Males
scholarshipsTable.filter(scholarshipsTable.Sex == '1') \
                .select('scholarships_id', 'Sex') \
                .dropDuplicates() \
                .count()
Out[31]: 910
```

```
In [25]: query = scholarshipsTable.select(["*"])\
                .join(countriesTable, (countriesTable.country_id == scholarshipsTable.country_id), how='inner')\
                .join(specializationTable, (specializationTable.spec_id == scholarshipsTable.spec_id), how='inner')\
                .select(['scholarships_id' ,specializationTable.spec_desc.alias('Specialization Name'),countriesTable.country_name.alias('Country Name'),countriesTable.Population.alias('Country Population'),\
                    'Sex', 'Educational_level','Academic_year','count'])\
                .filter(countriesTable.country_name != ('Asian Countries'))\
                .filter(countriesTable.country_name != ('European Countries'))\
                .filter(countriesTable.country_name != ('Arab Countries'))
query.toPandas().head(50)
```

Out[25]:

	scholarships_id	Specialization Name	Country Name	Country Population	Sex	Educational_level	Academic_year	count
0	170	Business, Management and Law	United States	298444215	2	Other	2015.0	1.0
1	171	Business, Management and Law	United States	298444215	1	Other	2015.0	4.0
2	172	Education	United States	298444215	2	Other	2015.0	1.0
3	173	Education	United States	298444215	1	Other	2015.0	6.0
4	174	Health and well-being	United States	298444215	1	Other	2015.0	17.0
5	175	Arts and humanities	United States	298444215	2	Other	2015.0	8.0
6	176	Arts and humanities	United States	298444215	1	Other	2015.0	2.0
7	177	Engineering, manufacturing and construction	United States	298444215	1	Other	2015.0	1.0
8	178	Communication_IT	United States	298444215	2	Other	2015.0	1.0
9	179	Communication_IT	United States	298444215	1	Other	2015.0	2.0
10	180	Not Specified	United States	298444215	2	Other	2015.0	1.0
11	181	Not Specified	United States	298444215	1	Other	2015.0	5.0
12	182	Health and well-being	United States	298444215	2	Fellowship	2015.0	98.0
13	183	Health and well-being	United States	298444215	1	Fellowship	2015.0	141.0
14	184	Not Specified	United States	298444215	2	Fellowship	2015.0	2.0
15	185	Business, Management and Law	United States	298444215	2	Bachelor	2015.0	208.0
16	186	Business, Management and Law	United States	298444215	1	Bachelor	2015.0	1287.0
17	187	Non-specialized programs and qualifications	United States	298444215	1	Bachelor	2015.0	1.0
18	188	Education	United States	298444215	2	Bachelor	2015.0	29.0
19	189	Education	United States	298444215	1	Bachelor	2015.0	13.0
20	190	Services	United States	298444215	2	Bachelor	2015.0	7.0
21	191	Services	United States	298444215	1	Bachelor	2015.0	47.0
22	192	Health and well-being	United States	298444215	2	Bachelor	2015.0	305.0
23	193	Health and well-being	United States	298444215	1	Bachelor	2015.0	298.0
24	194	Social Sciences, Journalism and Media	United States	298444215	2	Bachelor	2015.0	52.0
25	195	Social Sciences, Journalism and Media	United States	298444215	1	Bachelor	2015.0	190.0
26	196	Natural sciences, mathematics and statistics	United States	298444215	2	Bachelor	2015.0	139.0
27	197	Natural sciences, mathematics and statistics	United States	298444215	1	Bachelor	2015.0	289.0
28	198	Arts and humanities	United States	298444215	2	Bachelor	2015.0	125.0
29	199	Arts and humanities	United States	298444215	1	Bachelor	2015.0	88.0
30	200	Engineering, manufacturing and construction	United States	298444215	2	Bachelor	2015.0	129.0
31	201	Engineering, manufacturing and construction	United States	298444215	1	Bachelor	2015.0	1677.0
32	202	Communication_IT	United States	298444215	2	Bachelor	2015.0	83.0
33	203	Communication_IT	United States	298444215	1	Bachelor	2015.0	642.0
34	204	Not Specified	United States	298444215	2	Bachelor	2015.0	14.0
35	205	Not Specified	United States	298444215	1	Bachelor	2015.0	39.0
36	206	Business, Management and Law	United States	298444215	2	Ph.d	2015.0	26.0
37	207	Business, Management and Law	United States	298444215	1	Ph.d	2015.0	37.0
38	208	Education	United States	298444215	2	Ph.d	2015.0	12.0
39	209	Education	United States	298444215	1	Ph.d	2015.0	21.0
40	210	Health and well-being	United States	298444215	2	Ph.d	2015.0	19.0
41	211	Health and well-being	United States	298444215	1	Ph.d	2015.0	9.0
42	212	Social Sciences, Journalism and Media	United States	298444215	2	Ph.d	2015.0	5.0
43	213	Social Sciences, Journalism and Media	United States	298444215	1	Ph.d	2015.0	2.0
44	214	Natural sciences, mathematics and statistics	United States	298444215	2	Ph.d	2015.0	20.0
45	215	Natural sciences, mathematics and statistics	United States	298444215	1	Ph.d	2015.0	13.0
46	216	Arts and humanities	United States	298444215	2	Ph.d	2015.0	7.0
47	217	Arts and humanities	United States	298444215	1	Ph.d	2015.0	1.0
48	218	Engineering, manufacturing and construction	United States	298444215	2	Ph.d	2015.0	1.0
49	219	Engineering, manufacturing and construction	United States	298444215	1	Ph.d	2015.0	21.0

```
In [ ]:
```