Soccer Data Analysis

In [8]:
```
# This Python 3 environment comes with many helpful analytics libraries installed
# It is defined by the kaggle/python docker image: https://github.com/kaggle/docker-python
# For example, here's several helpful packages to load in

import numpy as np # linear algebra
import pandas as pd # data processing, CSV file I/O (e.g. pd.read_csv)

# Input data files are available in the "../input/" directory.
# For example, running this (by clicking run or pressing Shift+Enter) will list all files under the input directory

import os
for dirname, _, filenames in os.walk('/kaggle/input'):
    for filename in filenames:
        print(os.path.join(dirname, filename))

# Any results you write to the current directory are saved as output.
```

```
/kaggle/input/soccer/database.sqlite
```

Import all necessary libraries.

In [9]:
```
import numpy as np # linear algebra
import pandas as pd # data processing, CSV file I/O (e.g. pd.read_csv)
import sqlite3
import matplotlib.pyplot as pl
```

Connect with database. Select tables.

In [10]:
```
cnx = sqlite3.connect('/kaggle/input/soccer/database.sqlite')
tables = pd.read_sql("""SELECT *
                        FROM sqlite_master
                        WHERE type='table';""", cnx)
tables
```

Out[10]:

	type	name	tbl_name	rootpage	sql
0	table	sqlite_sequence	sqlite_sequence	4	CREATE TABLE sqlite_sequence(name,seq)
1	table	Player_Attributes	Player_Attributes	11	CREATE TABLE "Player_Attributes" (\n\t`id`\tIN...
2	table	Player	Player	14	CREATE TABLE `Player` (\n\t`id`\tINTEGER PRIMA...
3	table	Match	Match	18	CREATE TABLE `Match` (\n\t`id`\tINTEGER PRIMAR...
4	table	League	League	24	CREATE TABLE `League` (\n\t`id`\tINTEGER PRIMA...
5	table	Country	Country	26	CREATE TABLE `Country` (\n\t`id`\tINTEGER PRIM...
6	table	Team	Team	29	CREATE TABLE "Team" (\n\t`id`\tINTEGER PRIMARY...
7	table	Team_Attributes	Team_Attributes	2	CREATE TABLE `Team_Attributes` (\n\t`id`\tINTE...

Starting analyze the data:

1. Find out what is the team goal trend during seasons.
   - Create and select main data from different tables
   - Present data by line diagram.

In [11]:
```
league_goals = pd.read_sql_query("""SELECT
                                    avg(Match.home_team_goal) as Goals,
                                    country.name
                                    ,season
                                FROM Match
                                left join Country on Match.country_id= Country.id
                                where name in ('Switzerland', 'Netherlands', 'Spain', 'Germany', 'Belgium' )
                                group by Country.Name, season
                                order by Goals desc;""", cnx)
league_goals
```

Out[11]:

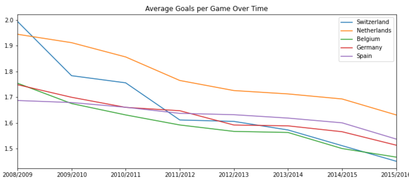
	Goals	name	season
0	1.994444	Switzerland	2009/2010
1	1.944444	Netherlands	2010/2011
2	1.911765	Netherlands	2011/2012
3	1.856209	Netherlands	2013/2014
4	1.783333	Switzerland	2015/2016
5	1.764706	Netherlands	2012/2013
6	1.755556	Switzerland	2008/2009
7	1.754167	Belgium	2011/2012
8	1.748366	Germany	2013/2014
9	1.725490	Netherlands	2009/2010
10	1.712418	Netherlands	2008/2009
11	1.699346	Germany	2008/2009
12	1.692810	Netherlands	2014/2015
13	1.686842	Spain	2012/2013
14	1.678947	Spain	2011/2012
15	1.675000	Belgium	2015/2016
16	1.660526	Spain	2008/2009
17	1.660131	Germany	2011/2012
18	1.647059	Germany	2010/2011
19	1.636842	Spain	2010/2011
20	1.631579	Spain	2013/2014
21	1.630719	Belgium	2008/2009
22	1.630719	Netherlands	2015/2016
23	1.618421	Spain	2015/2016
24	1.611111	Switzerland	2013/2014
25	1.605556	Switzerland	2014/2015
26	1.600000	Spain	2009/2010
27	1.591667	Belgium	2010/2011
28	1.591503	Germany	2012/2013
29	1.588235	Germany	2014/2015
30	1.572222	Switzerland	2010/2011
31	1.566667	Belgium	2014/2015
32	1.565359	Germany	2015/2016
33	1.562500	Belgium	2012/2013
34	1.536842	Spain	2014/2015
35	1.513072	Germany	2009/2010
36	1.511111	Switzerland	2012/2013
37	1.500000	Belgium	2013/2014
38	1.466667	Belgium	2009/2010
39	1.450617	Switzerland	2011/2012

In [12]:
```
df = pd.DataFrame(index=np.sort(league_goals['season'].unique()), columns=league_goals['name'].unique())

df.loc[:,'Spain'] = list(league_goals.loc[league_goals['name']=='Spain','Goals'])
df.loc[:,'Germany']   = list(league_goals.loc[league_goals['name']=='Germany','Goals'])
df.loc[:,'Belgium']   = list(league_goals.loc[league_goals['name']=='Belgium','Goals'])
df.loc[:,'Switzerland']   = list(league_goals.loc[league_goals['name']=='Switzerland','Goals'])
df.loc[:,'Netherlands']   = list(league_goals.loc[league_goals['name']=='Netherlands','Goals'])

df.plot(figsize=(12,5),title='Average Goals per Game Over Time')
```

Out[12]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x7fc5d8f72a58>
```



1. Is the difference how old player are in the countries analyzing seasons results?
   - From the results above chooce two countries (one with good results, another one - small amount of goals).
   - Create a query, which it filters for two countries.
   - Present data into visual.

In [13]:
```
detailed_matches = pd.read_sql("""SELECT
                                    country.name
                                    ,season
                                    ,avg(current_date- player.birthday) as age
                                FROM Match
                                left join Country on Match.country_id= Country.id
                                left join Player on Match.home_player_1 = Player.player_api_id
                                where name in ('Netherlands', 'Belgium' )

                                group by Country.Name,season
                                order by age desc;""", cnx)
detailed_matches
```

Out[13]:

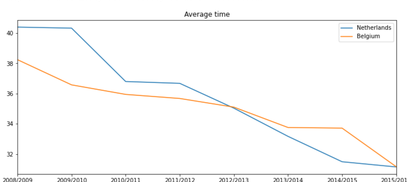
	name	season	age
0	Netherlands	2009/2010	40.394737
1	Netherlands	2008/2009	40.326316
2	Belgium	2008/2009	38.242105
3	Netherlands	2010/2011	36.797386
4	Netherlands	2011/2012	36.676471
5	Belgium	2009/2010	36.576190
6	Belgium	2011/2012	35.945607
7	Belgium	2010/2011	35.676471
8	Belgium	2012/2013	35.104167
9	Netherlands	2012/2013	35.036066
10	Belgium	2015/2016	33.758333
11	Belgium	2014/2015	33.716667
12	Netherlands	2013/2014	33.173203
13	Netherlands	2014/2015	31.493464
14	Belgium	2013/2014	31.166667
15	Netherlands	2015/2016	31.150327

In [14]:
```
df = pd.DataFrame(index=np.sort(detailed_matches['season'].unique()), columns=detailed_matches['name'].unique())

df.loc[:,'Belgium'] = list(detailed_matches.loc[detailed_matches['name']=='Belgium','age'])
df.loc[:,'Netherlands']   = list(detailed_matches.loc[detailed_matches['name']=='Netherlands','age'])

df.plot(figsize=(12,5),title='Average time')
```

Out[14]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x7fc5d8ebc6a0>
```



To sum up, from the data cuold be decided, that oldest players are playing more profesional and gain more goals than younger players.