내 구글 드라이브 마운팅

본인 구글 내 데이터 사용

```
1 from google.colab import drive
2 drive.mount('/content/drive')
```

```
Go to this URL in a browser: https://accounts.google.com/o/oauth2/auth?client_id=947318989803-6bn6qk8qdgf4n4g3pfee6491hc0brc4i.apps.googleuse

Enter your authorization code:
 · · · · · · · · · · ·
Mounted at /content/drive
```

선수정보, 연봉, 타자, 투수, 팀 데이터 읽어오기

```
1 import pandas as pd
2 player=pd.read_csv('/content/drive/My Drive/통계적방법19_박초연/People.csv')
3 salary=pd.read_csv('/content/drive/My Drive/통계적방법19_박초연/Salaries.csv')
4 batter=pd.read_csv('/content/drive/My Drive/통계적방법19_박초연/Batting.csv')
5 pitcher=pd.read_csv('/content/drive/My Drive/통계적방법19_박초연/Pitching.csv')
6 team=pd.read_csv('/content/drive/My Drive/통계적방법19_박초연/Teams.csv')
```

만든 데이터프레임 변수명 출력

```
1 print('선수정보',player.columns,'₩n연봉',salary.columns,
2 '₩n타자',batter.columns,'₩n투수',pitcher.columns,'₩n팀',team.columns)
```

```
선수정보 Index(['playerID', 'birthYear', 'birthMonth', 'birthDay', 'birthCountry',
       'birthState', 'birthCity', 'deathYear', 'deathMonth', 'deathDay',
       'deathCountry', 'deathState', 'deathCity', 'nameFirst', 'nameLast',
       'nameGiven', 'weight', 'height', 'bats', 'throws', 'debut', 'finalGame',
       'retroID', 'bbrefID'],
      dtype='object')
연봉 Index(['yearID', 'teamID', 'lgID', 'playerID', 'salary'], dtype='object')
타자 Index(['playerID', 'yearID', 'stint', 'teamID', 'lgID', 'G', 'AB', 'R', 'H',
       '2B', '3B', 'HR', 'RBI', 'SB', 'CS', 'BB', 'SO', 'IBB', 'HBP', 'SH',
       'SF', 'GIDP'],
      dtype='object')
투수 Index(['playerID', 'yearID', 'stint', 'teamID', 'lgID', 'W', 'L', 'G', 'GS',
       'CG', 'SHO', 'SV', 'IPouts', 'H', 'ER', 'HR', 'BB', 'SO', 'BAOpp',
       'ERA', 'IBB', 'WP', 'HBP', 'BK', 'BFP', 'GF', 'R', 'SH', 'SF', 'GIDP'],
      dtype='object')
팀 Index(['yearID', 'lgID', 'teamID', 'franchID', 'divID', 'Rank', 'G', 'Ghome',
       'W', 'L', 'DivWin', 'WCWin', 'LgWin', 'WSWin', 'R', 'AB', 'H', '2B',
       '3B', 'HR', 'BB', 'SO', 'SB', 'CS', 'HBP', 'SF', 'RA', 'ER', 'ERA',
       'CG', 'SHO', 'SV', 'IPouts', 'HA', 'HRA', 'BBA', 'SOA', 'E', 'DP', 'FP',
       'name', 'park', 'attendance', 'BPF', 'PPF', 'teamIDBR',
       'teamIDlahman45', 'teamIDretro'],
      dtype='object')
```

만든 데이터프레임 크기 보기

```
print('선수정보',player.shape,'Wn연봉',salary.shape,'Wn타자',batter.shape,
'Wn투수',pitcher.shape,'Wn팀',team.shape)
```

```
선수정보 (19617, 24)
연봉 (26428, 5)
타자 (105861, 22)
투수 (46699, 30)
팀 (2895, 48)
```

선수정보 + 연봉정보 합치기 (선수_연봉)

합친 데이터프레임 이름 : (선수_연봉) ps

```
ps=pd.merge(salary,player,on='playerID',how='inner')
ps.sort_values(['yearID','playerID'],inplace=True)
print(ps.shape,'₩n',ps.columns)
```

```
(26428, 28)
 Index(['yearID', 'teamID', 'lgID', 'playerID', 'salary', 'birthYear',
       'birthMonth', 'birthDay', 'birthCountry', 'birthState', 'birthCity',
       'deathYear', 'deathMonth', 'deathDay', 'deathCountry', 'deathState',
       'deathCity', 'nameFirst', 'nameLast', 'nameGiven', 'weight', 'height',
       'bats', 'throws', 'debut', 'finalGame', 'retroID', 'bbrefID'],
      dtype='object')
```

타자 정보 + 선수_연봉정보 합치기

```
bat=pd.merge(ps,batter,on=['yearID','playerID'],how='inner')
print(bat.shape,'₩n',bat.columns)
```

```
(28294, 48)
 Index(['yearID', 'teamID_x', 'lgID_x', 'playerID', 'salary', 'birthYear',
       'birthMonth', 'birthDay', 'birthCountry', 'birthState', 'birthCity',
       'deathYear', 'deathMonth', 'deathDay', 'deathCountry', 'deathState',
       'deathCity', 'nameFirst', 'nameLast', 'nameGiven', 'weight', 'height',
       'bats', 'throws', 'debut', 'finalGame', 'retroID', 'bbrefID', 'stint',
       'teamID_y', 'lgID_y', 'G', 'AB', 'R', 'H', '2B', '3B', 'HR', 'RBI',
       'SB', 'CS', 'BB', 'SO', 'IBB', 'HBP', 'SH', 'SF', 'GIDP'],
      dtype='object')
```

투수 정보 + 선수_연봉정보 합치기

```
pitch=pd.merge(ps,pitcher,on=['yearID','playerID'],how='inner')
print(pitch.shape,'₩n',pitch.columns)
```

```
2 print(pitch.shape, 'ㅠㅠ',pitch.columns)
```

```
(13365, 56)
Index(['yearID', 'teamID_x', 'lgID_x', 'playerID', 'salary', 'birthYear',
       'birthMonth', 'birthDay', 'birthCountry', 'birthState', 'birthCity',
       'deathYear', 'deathMonth', 'deathDay', 'deathCountry', 'deathState',
       'deathCity', 'nameFirst', 'nameLast', 'nameGiven', 'weight', 'height',
       'bats', 'throws', 'debut', 'finalGame', 'retroID', 'bbrefID', 'stint',
       'teamID_y', 'lgID_y', 'W', 'L', 'G', 'GS', 'CG', 'SHO', 'SV', 'IPouts',
       'H', 'ER', 'HR', 'BB', 'SO', 'BAOpp', 'ERA', 'IBB', 'WP', 'HBP', 'BK',
       'BFP', 'GF', 'R', 'SH', 'SF', 'GIDP'],
      dtype='object')
```

```
1 pitch.head(10)
```

	yearID	teamID_x	lgID_x	playerID	salary	birthYear	birthMonth	birthDay	birthCountry	birthState	birthCity	deathYe
0	1985	TOR	AL	ackerji01	170000	1958.0	9.0	24.0	USA	TX	Freer	N
1	1985	CHA	AL	agostju01	147500	1958.0	2.0	23.0	P.R.	NaN	Rio Piedras	N
2	1985	TOR	AL	alexado01	875000	1950.0	9.0	4.0	USA	AL	Cordova	N
3	1985	SLN	NL	allenne01	750000	1958.0	1.0	24.0	USA	KS	Kansas City	N
4	1985	SLN	NL	allenne01	750000	1958.0	1.0	24.0	USA	KS	Kansas City	N
5	1985	PHI	NL	anderla02	250500	1953.0	5.0	6.0	USA	OR	Portland	N
6	1985	SLN	NL	andujjo01	1030000	1952.0	12.0	21.0	D.R.	San Pedro de Macoris	San Pedro de Macoris	201
7	1985	OAK	AL	atherke01	107333	1959.0	2.0	19.0	USA	VA	Newport News	N
8	1985	CHA	AL	bannifl01	811250	1955.0	6.0	10.0	USA	SD	Pierre	N
9	1985	ATL	NL	barkele01	870000	1955.0	7.0	7.0	USA	KY	Fort Knox	N

[HW#1 due 내일 수업전까지]

bat, pitch 데이터프레임에서 stint=2인 선수(시즌 중 이적한 선수임)를 제외하는 코드를 작성하시오.

↳ 숨겨진 셀 11개

#1017@

11.04.화

과업1

2016년 데이터 가져오기

```
1 bat16 = bat2[bat2.yearID == 2016]
2 bat16.head(5)
```

	yearID	teamID_x	lgID_x	playerID	salary	birthYear	birthMonth	birthDay	birthCountry	birthState	birthCity
27395	2016	CHA	AL	abreujo02	11666667	1987.0	1.0	29.0	Cuba	Cienfuegos	Cienfuegos
27396	2016	NYA	AL	ackledu01	3200000	1988.0	2.0	26.0	USA	NC	Winston-Salem
27397	2016	COL	NL	adamecr01	509500	1991.0	7.0	26.0	D.R.	Distrito Nacional	Santo Domingo
27398	2016	SLN	NL	adamsma01	1650000	1988.0	8.0	31.0	USA	PA	Philipsburg
27399	2016	SFN	NL	adriaeh01	512500	1989.0	8.0	21.0	Venezuela	Miranda	Guarenas

과업 2

LB와 AL의 연봉의 차이가 있는지 검정하시오.

```
#등분산검정
#귀무가설 : NL과 AL의 분산은 같다.
#대립가설 : NL과 AL의 분산은 다르다.(양측검정)
import scipy
from scipy import stats
stats.levene(bat16[bat16.lgID_x=='NL'].salary,bat16[bat16.lgID_x=='AL'].salary)
```

```
LeveneResult(statistic=3.6022435285539287, pvalue=0.05809859930964574)
```

유의확률이 0.05보다 크므로 귀무가설(등분산) 채택. 등분산이다.

귀무가설 : NL과 AL의 연봉은 같다.

대립가설 : NL과 AL의 연봉은 다르다.(양측검정)

```
from scipy import stats
stats.ttest_ind(bat16[bat16.lgID_x=='NL'].salary,bat16[bat16.lgID_x=='AL'].salary,equal_var=True)
```

```
Ttest_indResult(statistic=-1.85527716943194, pvalue=0.06396236065581924)
```

결론 : 유의확률이 0.05보다 크므로 귀무가설 채택. 실제 평균은 다르지만 통계적으로는 연봉 차이가 없다.

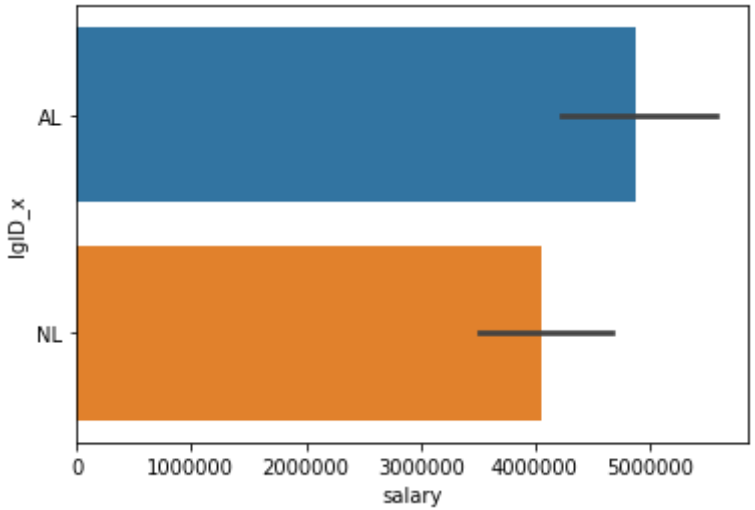
```
bat16.groupby(['lgID_x'])['salary'].mean()
```

```
lgID_x
AL    4.880440e+06
NL    4.050789e+06
Name: salary, dtype: float64
```

과업3

리그별 연봉 평균, 표준편차 그리기

```
1 import seaborn as sns
2 import matplotlib.pyplot as plt
3
4 sns.barplot(x=bat16.salary, y=bat16.lgID_x, data = bat16)
5 plt.show()
```



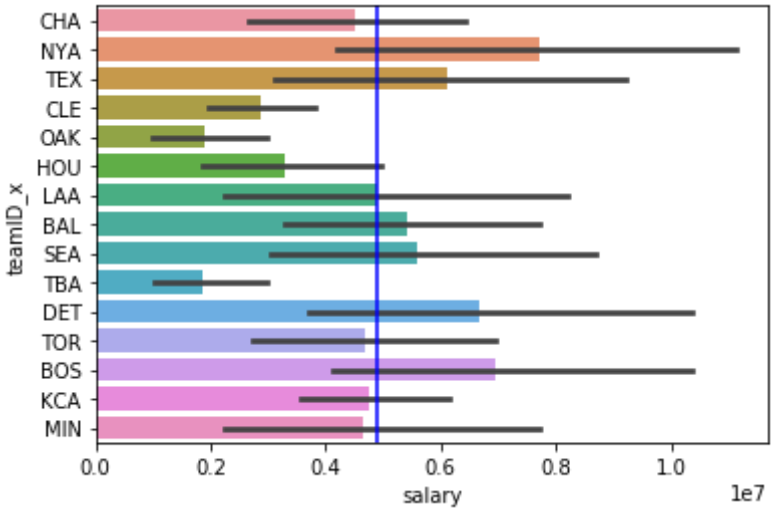
과업4

- AL 소속 선수만 선택하여 bat16al에 저장하시오.
- 팀별로 위의 그래프를 그리시오.
- AL 선수 연봉 평균으로 수직참조선을 그리시오.

```
1 bat16al = bat16[bat16.lgID_x=='AL']
```

```
1 import seaborn as sns
2 import matplotlib.pyplot as plt
3
```

```
4 sns.barplot(x=bat16al.salary, y=bat16al.teamID_x, data = bat16al)
5 plt.axvline(x= bat16al.salary.mean(), color= 'b' , linestyle= '-')
6
7 plt.show()
```

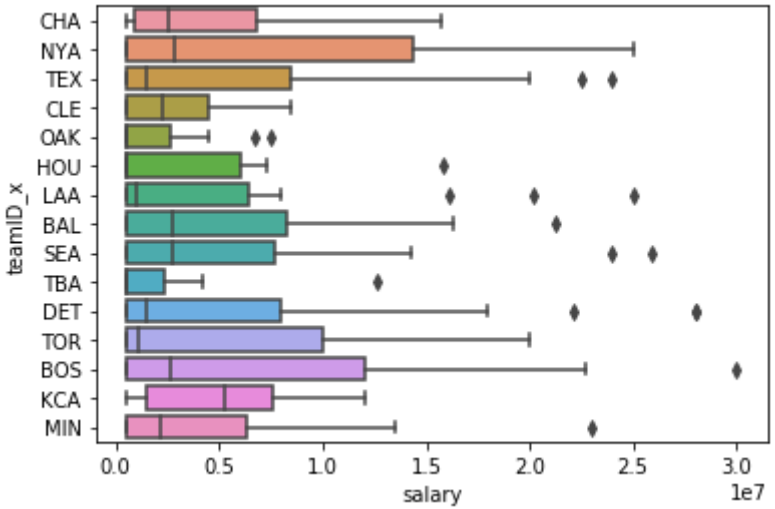


과업5

팀별 선수연봉 나무상자그림을 그리시오.

```
1 import seaborn as sns
2 sns.boxplot(x=bat16al.salary, y=bat16al.teamID_x,data=bat16al)
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f75431342e8>
```



과업6

> 팀별 선수연봉 평균, 표준편차, 95% 신뢰구간을 구하시오.

```
1 !pip install researchpy
```

```
Requirement already satisfied: researchpy in /usr/local/lib/python3.6/dist-packages (0.1.7)
Requirement already satisfied: pandas in /usr/local/lib/python3.6/dist-packages (from researchpy) (0.25.2)
Requirement already satisfied: numpy in /usr/local/lib/python3.6/dist-packages (from researchpy) (1.17.3)
Requirement already satisfied: scipy in /usr/local/lib/python3.6/dist-packages (from researchpy) (1.3.1)
Requirement already satisfied: statsmodels in /usr/local/lib/python3.6/dist-packages (from researchpy) (0.10.1)
Requirement already satisfied: python-dateutil>=2.6.1 in /usr/local/lib/python3.6/dist-packages (from pandas->researchpy) (2.6.1)
Requirement already satisfied: pytz>=2017.2 in /usr/local/lib/python3.6/dist-packages (from pandas->researchpy) (2018.9)
Requirement already satisfied: patsy>=0.4.0 in /usr/local/lib/python3.6/dist-packages (from statsmodels->researchpy) (0.5.1)
Requirement already satisfied: six>=1.5 in /usr/local/lib/python3.6/dist-packages (from python-dateutil>=2.6.1->pandas->researchpy) (1.12.0)
```

```
1 import researchpy as rp
2 rp.summary_cont(bat16al.salary.groupby(bat16al.teamID_x))  # SE = SD / 루트N
```

teamID_x	N	Mean	SD	SE	95% Conf.	Interval
BAL	26	5.427453e+06	5.913707e+06	1.159773e+06	3.109280e+06	7.745626e+06
BOS	27	6.942315e+06	8.382771e+06	1.613265e+06	3.720082e+06	1.016455e+07
CHA	23	4.521681e+06	4.733344e+06	9.869705e+05	2.543743e+06	6.499620e+06
CLE	25	2.852476e+06	2.485918e+06	4.971836e+05	1.857902e+06	3.847050e+06
DET	29	6.684844e+06	8.785620e+06	1.631449e+06	3.430605e+06	9.939083e+06
HOU	23	3.288413e+06	3.817006e+06	7.959008e+05	1.693388e+06	4.883438e+06
KCA	27	4.741190e+06	3.384641e+06	6.513745e+05	3.440176e+06	6.042204e+06
LAA	21	4.891730e+06	7.088649e+06	1.546870e+06	1.784993e+06	7.998468e+06
MIN	17	4.658306e+06	6.079803e+06	1.474569e+06	1.679202e+06	7.637409e+06
NYA	24	7.704335e+06	9.036625e+06	1.844593e+06	4.011173e+06	1.139750e+07
OAK	17	1.898063e+06	2.276473e+06	5.521258e+05	7.825910e+05	3.013535e+06
SEA	22	5.600288e+06	7.365644e+06	1.570361e+06	2.449950e+06	8.750626e+06
TBA	25	1.842492e+06	2.578162e+06	5.156323e+05	8.110129e+05	2.873972e+06
TEX	24	6.094176e+06	7.838800e+06	1.600088e+06	2.890551e+06	9.297801e+06
TOR	27	4.678767e+06	5.853185e+06	1.126446e+06	2.428875e+06	6.928659e+06

과업7

귀무가설 : 모든 팀의 선수연봉은 동일하다.

mu1=mu2=mu3

대립가설 : 적어도 한 팀의 선수연봉은 다르다

```
import statsmodels.api as sm
from statsmodels.formula.api import ols
results = ols('salary~teamID_x',data=bat16al).fit()
results.summary()
```

OLS Regression Results

Dep. Variable:	salary	**R-squared:**	0.070
Model:	OLS	**Adj. R-squared:**	0.032
Method:	Least Squares	**F-statistic:**	1.846
Date:	Tue, 05 Nov 2019	**Prob (F-statistic):**	0.0313
Time:	05:25:37	**Log-Likelihood:**	-6084.2
No. Observations:	357	**AIC:**	1.220e+04
Df Residuals:	342	**BIC:**	1.226e+04
Df Model:	14		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
Intercept	5.427e+06	1.22e+06	4.441	0.000	3.02e+06	7.83e+06
teamID_x[T.BOS]	1.515e+06	1.71e+06	0.885	0.377	-1.85e+06	4.88e+06
teamID_x[T.CHA]	-9.058e+05	1.78e+06	-0.508	0.612	-4.41e+06	2.6e+06
teamID_x[T.CLE]	-2.575e+06	1.75e+06	-1.475	0.141	-6.01e+06	8.59e+05
teamID_x[T.DET]	1.257e+06	1.68e+06	0.747	0.456	-2.05e+06	4.57e+06
teamID_x[T.HOU]	-2.139e+06	1.78e+06	-1.199	0.231	-5.65e+06	1.37e+06
teamID_x[T.KCA]	-6.863e+05	1.71e+06	-0.401	0.689	-4.05e+06	2.68e+06
teamID_x[T.LAA]	-5.357e+05	1.83e+06	-0.293	0.770	-4.13e+06	3.06e+06
teamID_x[T.MIN]	-7.691e+05	1.94e+06	-0.396	0.693	-4.59e+06	3.05e+06
teamID_x[T.NYA]	2.277e+06	1.76e+06	1.291	0.198	-1.19e+06	5.75e+06
teamID_x[T.OAK]	-3.529e+06	1.94e+06	-1.816	0.070	-7.35e+06	2.94e+05
teamID_x[T.SEA]	1.728e+05	1.81e+06	0.096	0.924	-3.38e+06	3.72e+06
teamID_x[T.TBA]	-3.585e+06	1.75e+06	-2.054	0.041	-7.02e+06	-1.51e+05
teamID_x[T.TEX]	6.667e+05	1.76e+06	0.378	0.706	-2.8e+06	4.14e+06
teamID_x[T.TOR]	-7.487e+05	1.71e+06	-0.437	0.662	-4.12e+06	2.62e+06

Omnibus:	95.758	**Durbin-Watson:**	2.004
Prob(Omnibus):	0.000	**Jarque-Bera (JB):**	182.225
Skew:	1.468	**Prob(JB):**	2.69e-40
Kurtosis:	4.904	**Cond. No.**	15.3

Warnings:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.

```
1 aov_table=sm.stats.anova_lm(results, typ=2) # df = 집단수 - 1
2 aov_table
```

	sum_sq	df	F	PR(>F)
teamID_x	1.003597e+15	14.0	1.845763	0.031322
Residual	1.328255e+16	342.0	NaN	NaN

과업 8

> AL과 NL을 선수엽봉차이에 대한 분산분석하시오.

```
1 results = ols('salary~lgID_x',data=bat16).fit()
2 results.summary()
```

OLS Regression Results

Dep. Variable:	salary	**R-squared:**	0.005
Model:	OLS	**Adj. R-squared:**	0.003
Method:	Least Squares	**F-statistic:**	3.442
Date:	Tue, 05 Nov 2019	**Prob (F-statistic):**	0.0640
Time:	05:24:30	**Log-Likelihood:**	-12380.
No. Observations:	727	**AIC:**	2.476e+04
Df Residuals:	725	**BIC:**	2.477e+04
Df Model:	1		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
Intercept	4.88e+06	3.19e+05	15.298	0.000	4.25e+06	5.51e+06
lgID_x[T.NL]	-8.297e+05	4.47e+05	-1.855	0.064	-1.71e+06	4.83e+04

Omnibus:	284.427	**Durbin-Watson:**	2.039
Prob(Omnibus):	0.000	**Jarque-Bera (JB):**	868.135
Skew:	1.967	**Prob(JB):**	3.07e-189
Kurtosis:	6.630	**Cond. No.**	2.64

Warnings:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.

```
1 aov_table=sm.stats.anova_lm(results, typ=2)
2 aov_table
```

	sum_sq	df	F	PR(>F)
lgID_x	1.250623e+14	1.0	3.442053	0.063962
Residual	2.634188e+16	725.0	NaN	NaN

마지막 과업

팀별 선수연봉 쌍체비교를 Tukey 방법으로

```
from statsmodels.stats.multicomp import pairwise_tukeyhsd
from statsmodels.stats.multicomp import MultiComparison
mc=MultiComparison(bat16al.salary, bat16al.teamID_x)
print(mc.tukeyhsd())
```

Multiple Comparison of Means - Tukey HSD, FWER=0.05

group1	group2	meandiff	p-adj	lower	upper	reject
BAL	BOS	1514862.0684	0.9	-4335531.4039	7365255.5406	False
BAL	CHA	-905771.9799	0.9	-7000623.7122	5189079.7523	False
BAL	CLE	-2574977.1538	0.9	-8539066.0327	3389111.725	False
BAL	DET	1257391.0186	0.9	-4493185.2651	7007967.3023	False
BAL	HOU	-2139040.1104	0.9	-8233891.8426	3955811.6219	False
BAL	KCA	-686263.339	0.9	-6536656.8113	5164130.1332	False
BAL	LAA	-535723.011	0.9	-6782676.9639	5711230.9419	False
BAL	MIN	-769147.2715	0.9	-7410228.4694	5871933.9264	False
BAL	NYA	2276882.2212	0.9	-3750217.8789	8303982.3212	False
BAL	OAK	-3529390.3891	0.886	-10170471.587	3111690.8087	False
BAL	SEA	172834.9825	0.9	-5995078.9001	6340748.8651	False
BAL	TBA	-3584960.7538	0.7347	-9549049.6327	2379128.125	False
BAL	TEX	666722.8045	0.9	-5360377.2955	6693822.9045	False
BAL	TOR	-748686.4872	0.9	-6599079.9594	5101706.9851	False
BOS	CHA	-2420634.0483	0.9	-8462274.8211	3621006.7245	False
BOS	CLE	-4089839.2222	0.537	-9999539.8698	1819861.4253	False
BOS	DET	-257471.0498	0.9	-5951620.0805	5436677.9809	False
BOS	HOU	-3653902.1787	0.7266	-9695542.9515	2387738.594	False
BOS	KCA	-2201125.4074	0.9	-7996063.6615	3593812.8467	False
BOS	LAA	-2050585.0794	0.9	-8245634.6578	4144464.499	False
BOS	MIN	-2284009.3399	0.9	-8876290.0579	4308271.3782	False
BOS	NYA	762020.1528	0.9	-5211265.4761	6735305.7817	False
BOS	OAK	-5044252.4575	0.3663	-11636533.1756	1548028.2606	False
BOS	SEA	-1342027.0859	0.9	-7457365.7725	4773311.6007	False
BOS	TBA	-5099822.8222	0.1798	-11009523.4698	809877.8253	False
BOS	TEX	-848139.2639	0.9	-6821424.8928	5125146.365	False
BOS	TOR	-2263548.5556	0.9	-8058486.8097	3531389.6986	False
CHA	CLE	-1669205.1739	0.9	-7821007.8042	4482597.4564	False
CHA	DET	2163162.9985	0.9	-3781872.4969	8108198.4939	False
CHA	HOU	-1233268.1304	0.9	-7511925.3981	5045389.1373	False
CHA	KCA	219508.6409	0.9	-5822132.1319	6261149.4137	False
CHA	LAA	370048.9689	0.9	-6056361.6262	6796459.5641	False
CHA	MIN	136624.7084	0.9	-6673535.2694	6946784.6863	False
CHA	NYA	3182654.2011	0.9	-3030256.1526	9395564.5548	False
CHA	OAK	-2623618.4092	0.9	-9433778.3871	4186541.5686	False

CHA	SEA	1078606.9625	0.9	-5270997.8371	7428211.762	False
CHA	TBA	-2679188.7739	0.9	-8830991.4042	3472613.8564	False
CHA	TEX	1572494.7844	0.9	-4640415.5693	7785405.1381	False
CHA	TOR	157085.4928	0.9	-5884555.28	6198726.2655	False
CLE	DET	3832368.1724	0.6077	-1978534.1292	9643270.474	False
CLE	HOU	435937.0435	0.9	-5715865.5868	6587739.6738	False
CLE	KCA	1888713.8148	0.9	-4020986.8327	7798414.4624	False
CLE	LAA	2039254.1429	0.9	-4263276.4325	8341784.7182	False
CLE	MIN	1805829.8824	0.9	-4887556.2144	8499215.9791	False
CLE	NYA	4851859.375	0.2954	-1232825.7918	10936544.5418	False
CLE	OAK	-954413.2353	0.9	-7647799.3321	5738972.8615	False
CLE	SEA	2747812.1364	0.9	-3476384.1673	8972008.44	False
CLE	TBA	-1009983.6	0.9	-7032260.0897	5012292.8897	False
CLE	TEX	3241699.9583	0.8831	-2842985.2085	9326385.1252	False
CLE	TOR	1826290.6667	0.9	-4083409.9809	7735991.3142	False
DET	HOU	-3396431.1289	0.7994	-9341466.6244	2548604.3665	False
DET	KCA	-1943654.3576	0.9	-7637803.3883	3750494.6731	False
DET	LAA	-1793114.0296	0.9	-7893987.991	4307759.9319	False
DET	MIN	-2026538.2901	0.9	-8530397.8273	4477321.2472	False
DET	NYA	1019491.2026	0.9	-4856065.3685	6895047.7736	False
DET	OAK	-4786781.4077	0.4375	-11290640.945	1717078.1296	False
DET	SEA	-1084556.0361	0.9	-7104472.237	4935360.1649	False
DET	TBA	-4842351.7724	0.2276	-10653254.074	968550.5292	False
DET	TEX	-590668.2141	0.9	-6466224.7851	5284888.357	False
DET	TOR	-2006077.5057	0.9	-7700226.5365	3688071.525	False
HOU	KCA	1452776.7713	0.9	-4588864.0014	7494417.5441	False
HOU	LAA	1603317.0994	0.9	-4823093.4957	8029727.6945	False
HOU	MIN	1369892.8389	0.9	-5440267.139	8180052.8167	False
HOU	NYA	4415922.3315	0.4963	-1796988.0222	10628832.6852	False
HOU	OAK	-1390350.2788	0.9	-8200510.2566	5419809.6991	False
HOU	SEA	2311875.0929	0.9	-4037729.7066	8661479.8924	False
HOU	TBA	-1445920.6435	0.9	-7597723.2738	4705881.9868	False
HOU	TEX	2805762.9149	0.9	-3407147.4388	9018673.2685	False
HOU	TOR	1390353.6232	0.9	-4651287.1496	7431994.396	False
KCA	LAA	150540.328	0.9	-6044509.2504	6345589.9064	False
KCA	MIN	-82883.9325	0.9	-6675164.6505	6509396.7856	False
KCA	NYA	2963145.5602	0.9	-3010140.0687	8936431.1891	False
KCA	OAK	-2843127.0501	0.9	-9435407.7682	3749153.668	False
KCA	SEA	859098.3215	0.9	-5256240.3651	6974437.0082	False
KCA	TBA	-2898697.4148	0.9	-8808398.0624	3011003.2327	False

KCA	TEX	1352986.1435	0.9	-4620299.4854	7326271.7724	False
KCA	TOR	-62423.1481	0.9	-5857361.4023	5732515.106	False
LAA	MIN	-233424.2605	0.9	-7180041.781	6713193.26	False
LAA	NYA	2812605.2321	0.9	-3549585.5159	9174795.9802	False
LAA	OAK	-2993667.3782	0.9	-9940284.8986	3952950.1423	False
LAA	SEA	708557.9935	0.9	-5787186.549	7204302.536	False
LAA	TBA	-3049237.7429	0.9	-9351768.3182	3253292.8325	False
LAA	TEX	1202445.8155	0.9	-5159744.9326	7564636.5635	False
LAA	TOR	-212963.4762	0.9	-6408013.0546	5982086.1022	False
MIN	NYA	3046029.4926	0.9	-3703562.8581	9795621.8434	False
MIN	OAK	-2760243.1176	0.9	-10063326.0805	4542839.8452	False
MIN	SEA	941982.254	0.9	-5933643.0229	7817607.5309	False
MIN	TBA	-2815813.4824	0.9	-9509199.5791	3877572.6144	False
MIN	TEX	1435870.076	0.9	-5313722.2748	8185462.4267	False
MIN	TOR	20460.7843	0.9	-6571819.9338	6612741.5024	False
NYA	OAK	-5806272.6103	0.1839	-12555864.961	943319.7405	False
NYA	SEA	-2104047.2386	0.9	-8388647.392	4180552.9147	False
NYA	TBA	-5861842.975	0.0731	-11946528.1418	222842.1918	False
NYA	TEX	-1610159.4167	0.9	-7756619.6207	4536300.7874	False
NYA	TOR	-3025568.7083	0.9	-8998854.3372	2947716.9206	False
OAK	SEA	3702225.3717	0.8707	-3173399.9053	10577850.6486	False
OAK	TBA	-55570.3647	0.9	-6748956.4615	6637815.7321	False
OAK	TEX	4196113.1936	0.6899	-2553479.1571	10945705.5444	False
OAK	TOR	2780703.902	0.9	-3811576.8161	9372984.62	False
SEA	TBA	-3757795.7364	0.7289	-9981992.04	2466400.5673	False
SEA	TEX	493887.822	0.9	-5790712.3314	6778487.9753	False
SEA	TOR	-921521.4697	0.9	-7036860.1563	5193817.2169	False
TBA	TEX	4251683.5583	0.5225	-1833001.6085	10336368.7252	False
TBA	TOR	2836274.2667	0.9	-3073426.3809	8745974.9142	False
TEX	TOR	-1415409.2917	0.9	-7388694.9206	4557876.3372	False

결론 : 분산분석 결과 요인 수준에 따른 팀별 평균 차이가 있어도 쌍체 비교에서는 유의한 쌍체 pairwise 없을 수 있음.

전체적으로는 팀별 선수연봉의 차이가 있고, 쌍체비교를 통해 팀별로는 차이가 없음을 알 수 있다.