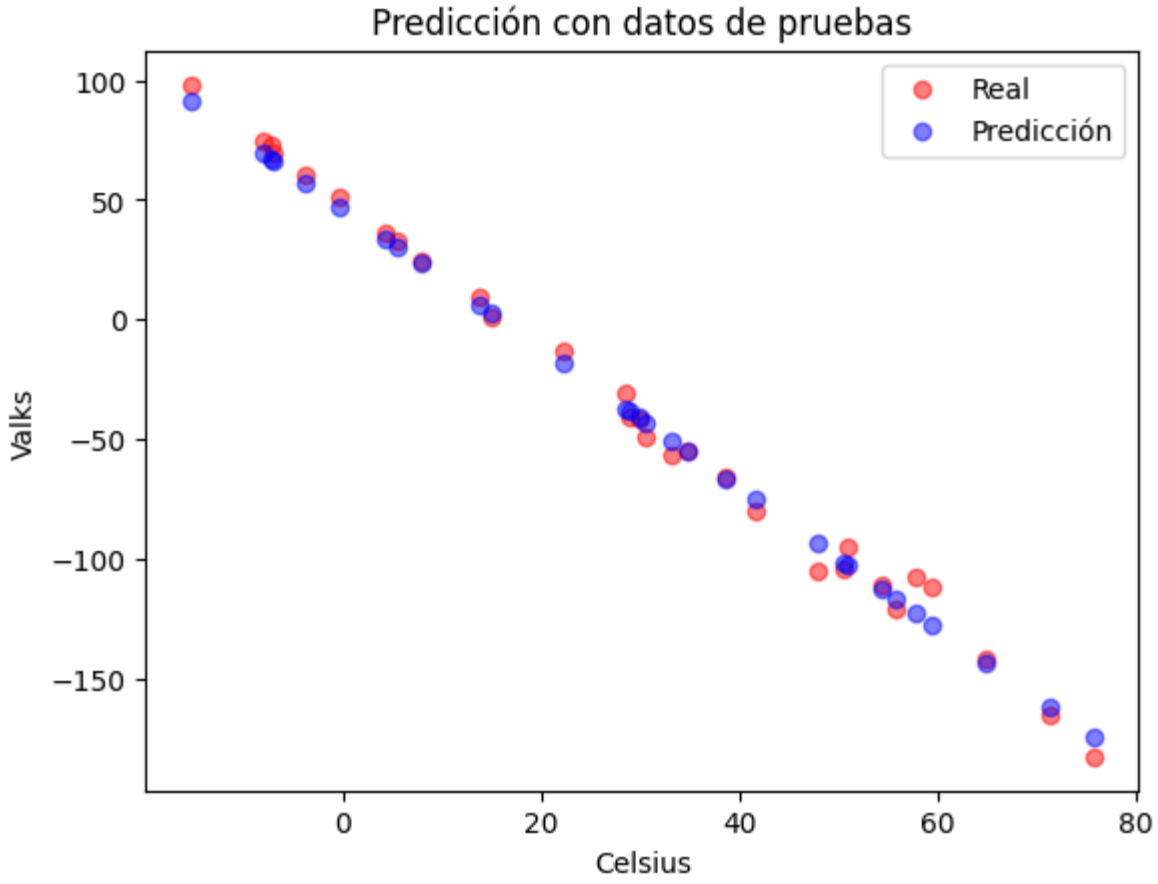
Momento de Retroalimentación: Módulo 2 Uso de framework o biblioteca de aprendizaje máquina para la implementación de una solución.

Oskar Arturo Gamboa Reyes | A01173648

In [2]:
```
import matplotlib.pyplot as plt
import pandas as pd
import numpy as np
from sklearn import linear_model
from sklearn.model_selection import train_test_split
from sklearn import metrics
colnames=['Celsius', 'Valks']
df = pd.read_csv('Valhalla23.csv', names=colnames, skiprows=1)
```

In [30]:
```
# Separar datos en subconjuntos (30% test 70% training)

x = df[['Celsius']]
y = df['Valks']

x_train, x_test, y_train, y_test = train_test_split(x, y, test_size=0.30, random_st

# Entrenar el modelo
# A partir de prueba y error pude determinar los hiper-parámetros, era necesario un
# y al usar una tasa tan baja tenia que tener un numero de iteraciones altas para q

model = linear_model.SGDRegressor(penalty=None, max_iter=1000000 ,eta0=1e-5, tol=1e
model.fit(x_train, y_train)

# Nuevas estimaciones

prediction_train = model.predict(x_train)
prediction_test = model.predict(x_test)

# Gráfica

plt.plot(x_test, y_test ,'ro', label="Real", alpha=0.5)
plt.plot(x_test, prediction_test, 'bo', label="Predicción" , alpha=0.5)
plt.title("Predicción con datos de pruebas")
plt.xlabel('Celsius')
plt.ylabel('Valks')
plt.legend()
plt.show()
```



In [31]:
```
print("Mean squared error train")
print(metrics.mean_squared_error(y_train, prediction_train))

print("Mean squared error test")
print(metrics.mean_squared_error(y_test, prediction_test))
```

```
Mean squared error train
54.567908443802494
Mean squared error test
38.04909852540704
```

In [33]:
```
!jupyter nbconvert --to html "/content/drive/MyDrive/ColabNotebooks/VallhalaScikit.
```

```
[NbConvertApp] Converting notebook /content/drive/MyDrive/ColabNotebooks/VallhalaSc
ikit.ipynb to html
[NbConvertApp] Writing 629232 bytes to /content/drive/MyDrive/ColabNotebooks/Vallha
laScikit.html
```