

# データプラットフォームと Pythonの出会い

## データ活用の新しいアプローチ

堀田 稔

インターシステムズジャパン株式会社

2023年4月20日



# アジェンダ

- 1 データ分析のためのアーキテクチャ - データプラットフォーム
- 2 サーバサイドプログラミング環境
- 3 Embedded Pythonについて
- 4 Embedded Pythonの機能概要
- 5 Embedded Pythonを学ぶためのリソース

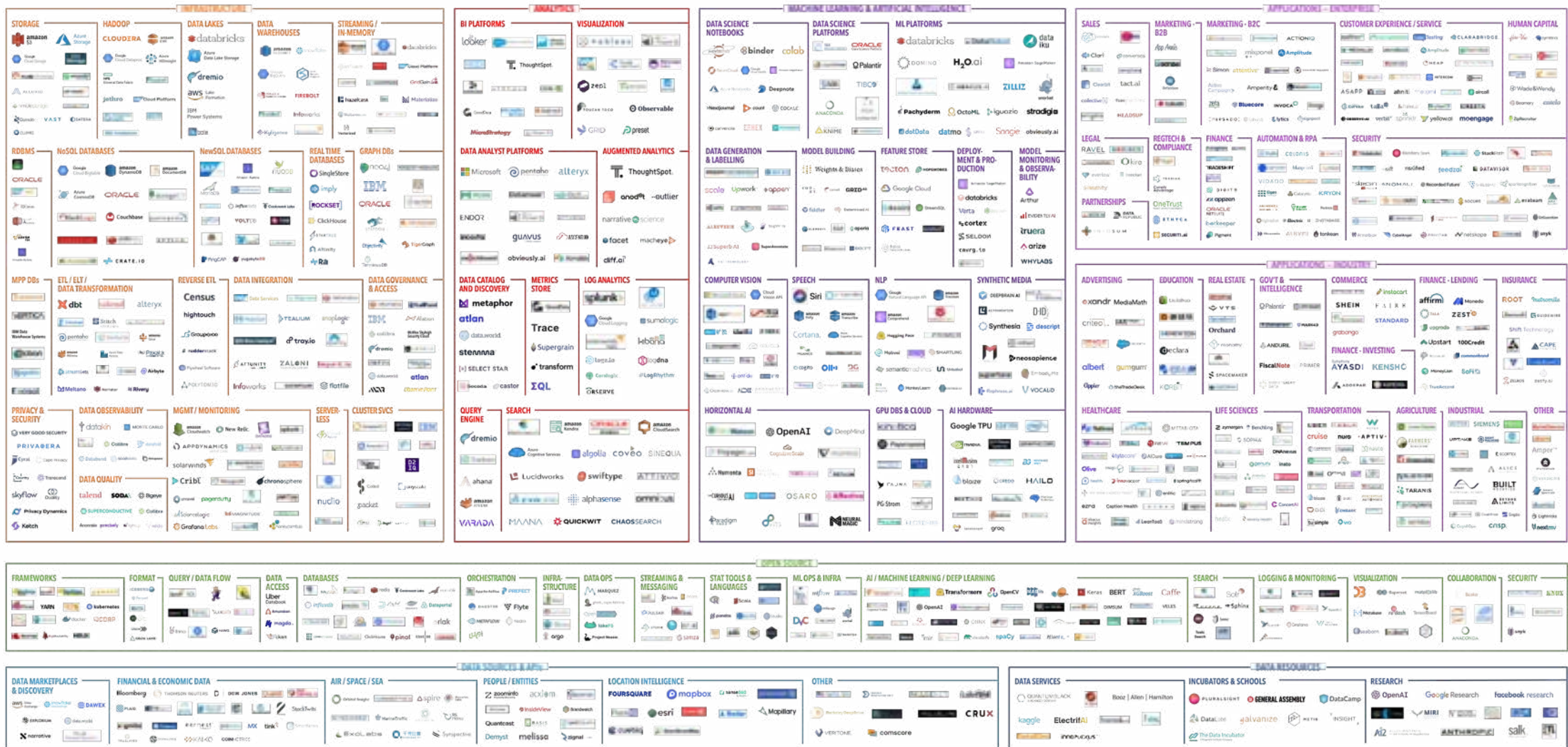


# データ分析に関連する歴史

- データウェアハウス
  - 業務システムで発生するデータは、業務システム上で分析できなかった。
  - 業務システムとは別にデータウェアハウスを構築し、分析ニーズに合わせてデータマートを作成した。
  - 業務システムとデータウェアハウス間にデータ変換ツール(ETL)が利用された。
  - 連携データ種の増加とともにデータウェアハウスのスキーマを修正せざるを得ない。
  - データの取り込みはバッチ処理が主流で、データの「鮮度」を保つのが困難。
- ビッグデータ、データレイク
  - 大量のデータを「とにかく溜める」
    - IoT、高頻度データ、データストリームへの対応の必要性
    - デジタルデータの爆発的増加(21世紀の石油)
  - ベストオブブリード(様々なツールの組み合わせ)は、アーキテクチャの設計や保守に高度なスキルが必要となる。
  - データガバナンスやSSoT(Single Source of Truth)の担保が難しく、データサイロが発生した。

# MACHINE LEARNING, ARTIFICIAL INTELLIGENCE, AND DATA (MAD) LANDSCAPE 2021

MACHINE LEARNING, ARTIFICIAL INTELLIGENCE, AND DATA (MAD) LANDSCAPE 2021



Version 3.0 - November 2021

© Matt Turck (@mattturck), John Wu (@john\_d\_wu) & FirstMark (@firstmarkcap)

mattturck.com/data2021

FIRSTMARK  
EARLY STAGE VENTURE CAPITAL

© Matt Turck (@mattturck), John Wu (@john\_d\_wu) & FirstMark (@firstmarkcap)



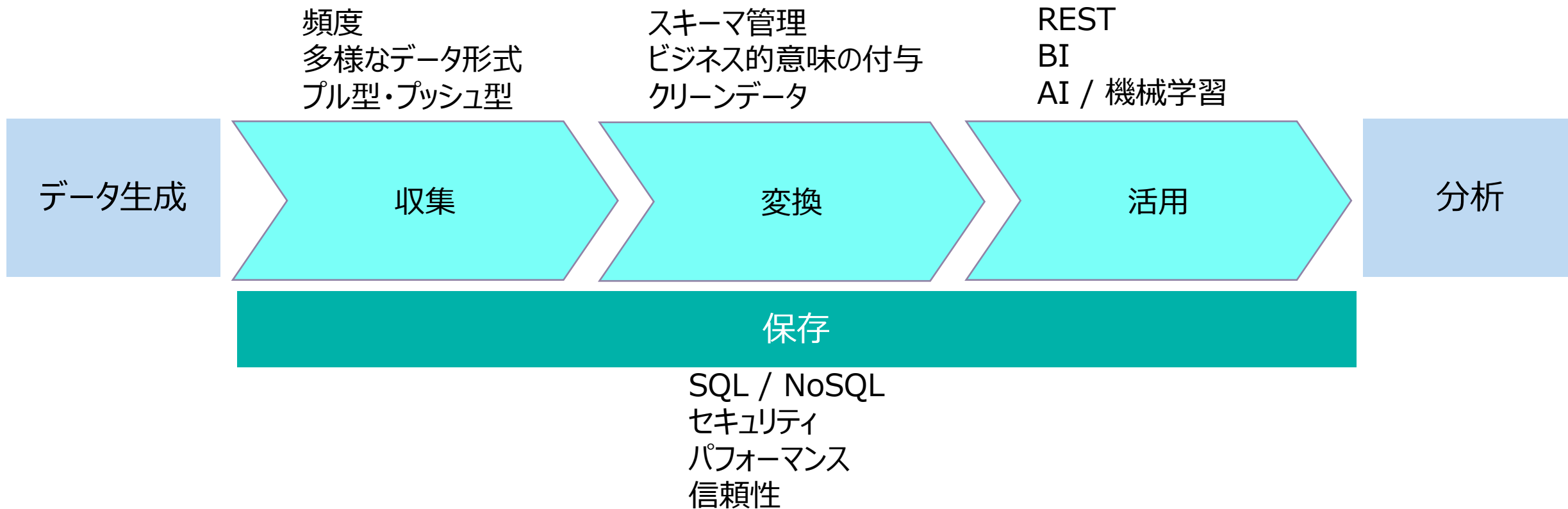
# データ分析に関連する歴史

- データウェアハウス
  - 業務システムで発生するデータは、業務システム上で分析できなかった。
  - 業務システムとは別にデータウェアハウスを構築し、分析ニーズに合わせてデータマートを作成した。
  - 業務システムとデータウェアハウス間にデータ変換ツール(ETL)が利用された。
  - 連携データ種の増加とともにデータウェアハウスのスキーマを修正せざるを得ない。
  - データの取り込みはバッチ処理が主流で、データの「鮮度」を保つのが困難。
- ビッグデータ、データレイク
  - 大量のデータを「とにかく溜める」
    - IoT、高頻度データ、データストリームへの対応の必要性
    - デジタルデータの爆発的増加(21世紀の石油)
  - ベストオブブリード(様々なツールの組み合わせ)は、アーキテクチャの設計や保守に高度なスキルが必要となる。
  - データガバナンスやSSoT(Single Source of Truth)の担保が難しく、データサイロ化が発生した。

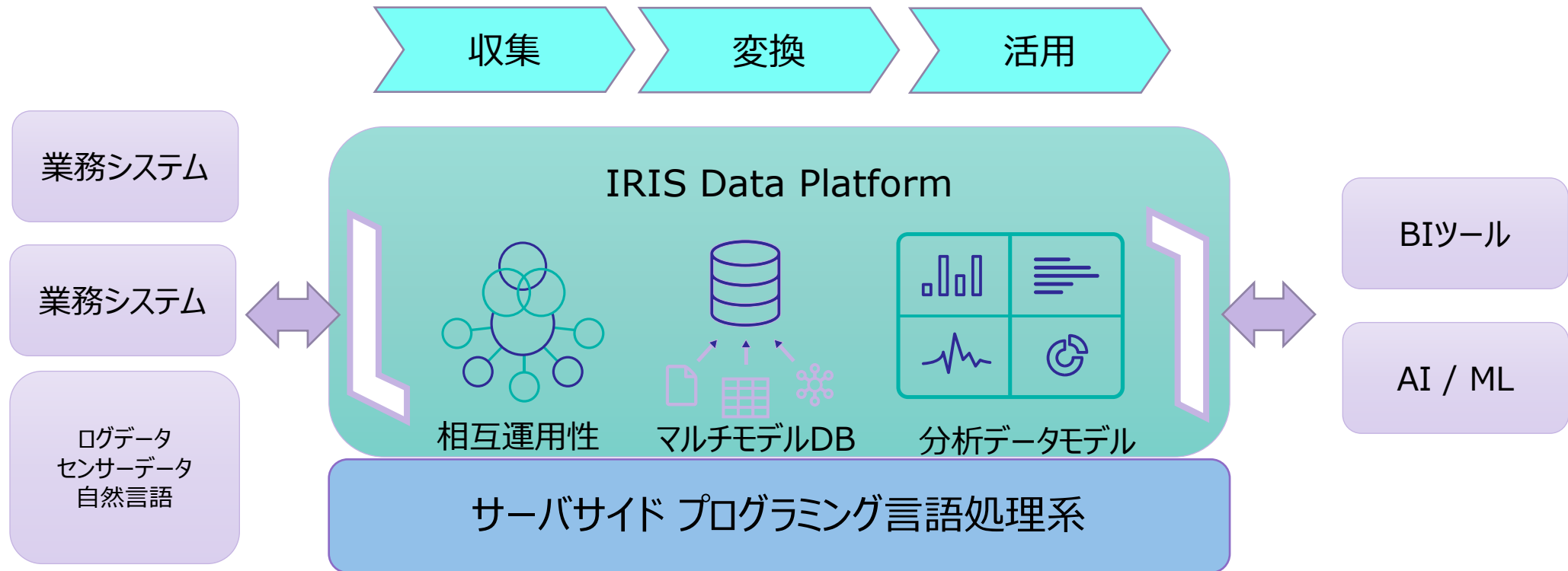


# データへのフォーカス：データエンジニアリングとデータプラットフォーム

- データの収集～活用に至るライフサイクルにフォーカスする「データエンジニアリング」
- データエンジニアリングの全てのフェーズ・技術に対応する「データプラットフォーム」



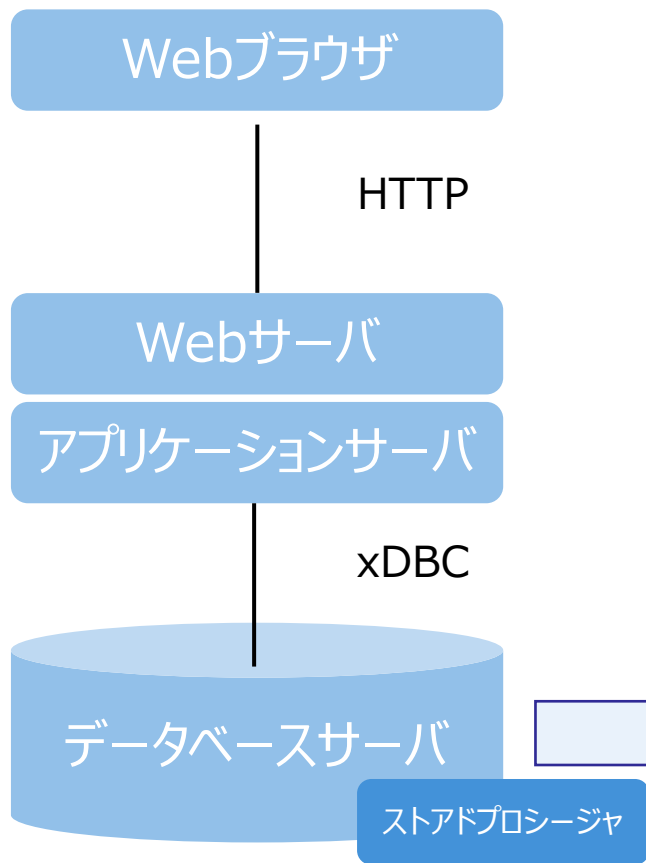
# InterSystems IRIS Data Platform



# 例) RESTアプリケーションのアーキテクチャ

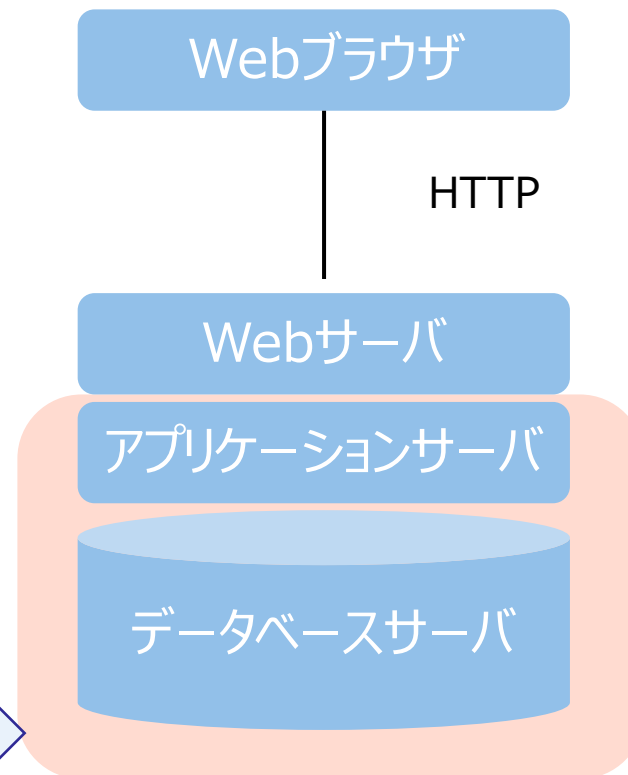


従来のアーキテクチャ



IRIS Data Platformのアーキテクチャ

シンプルな構成  
容易にスケーラビリティを確保  
インピーダンスミスマッチの回避

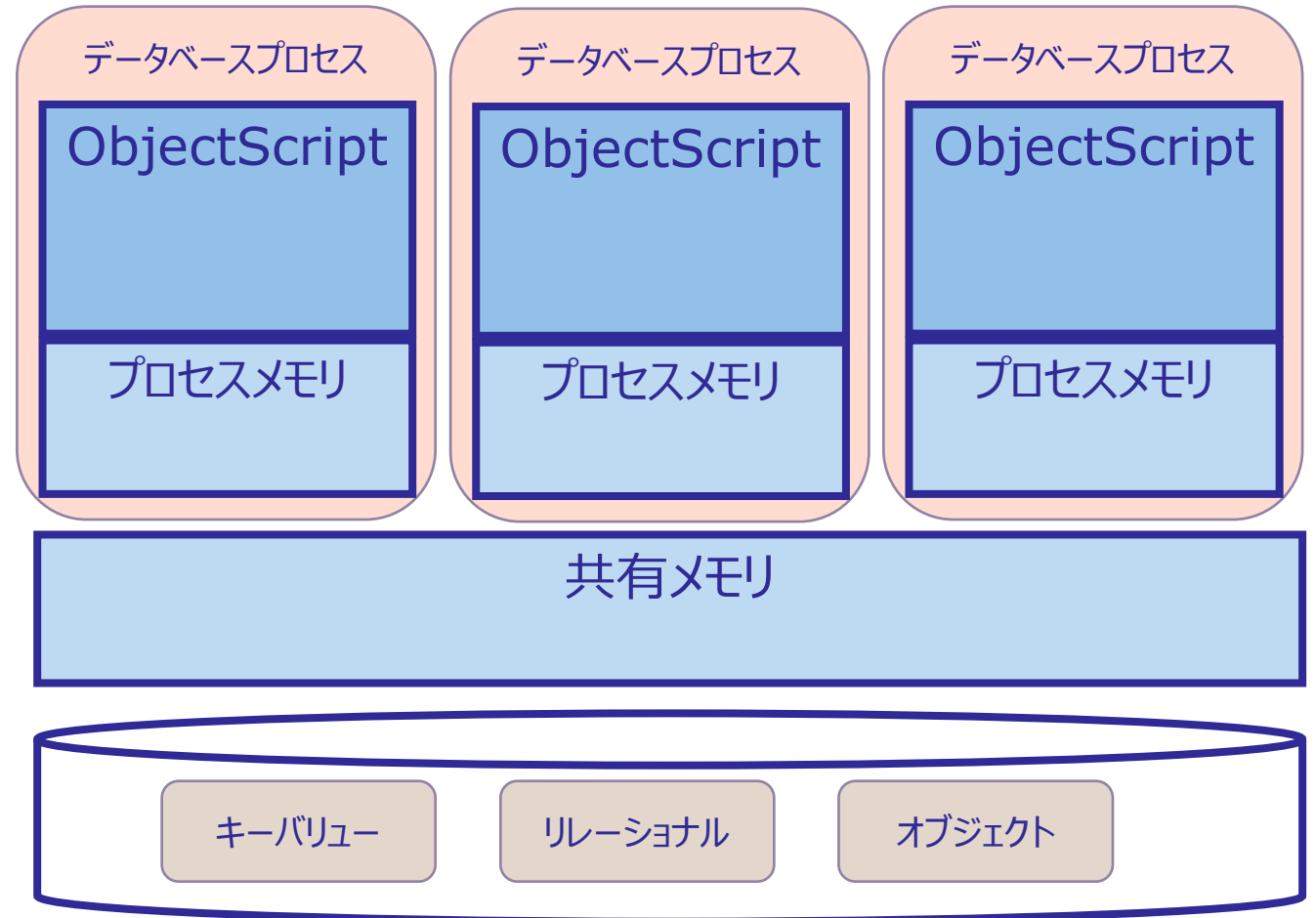




# ObjectScript



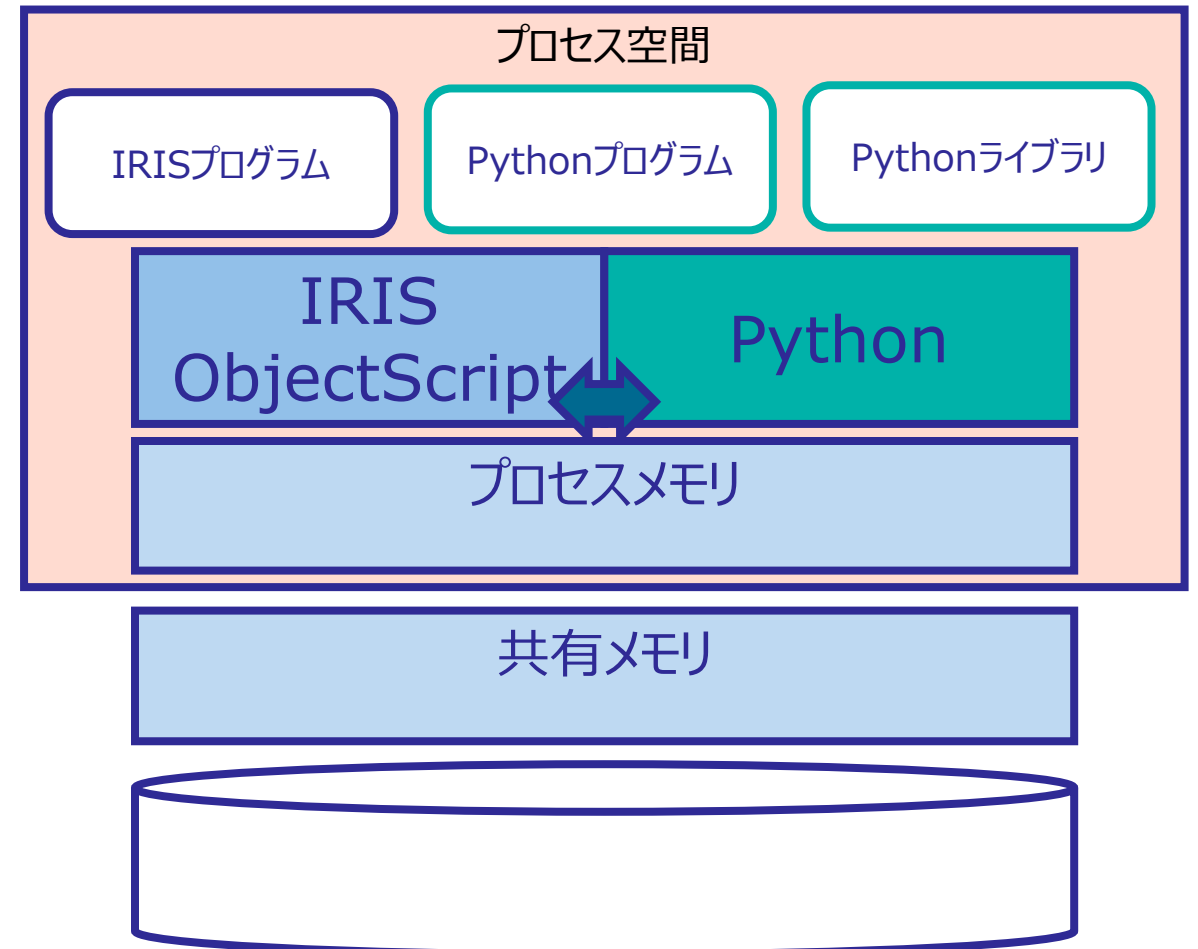
- データベース内のプロセスに、ObjectScriptの処理系を持つ
- IRISのマルチモデルDBに、共有メモリを介してフルアクセス可能
  - オブジェクト
  - SQL
  - キーバリュー
  - ドキュメント(JSON)



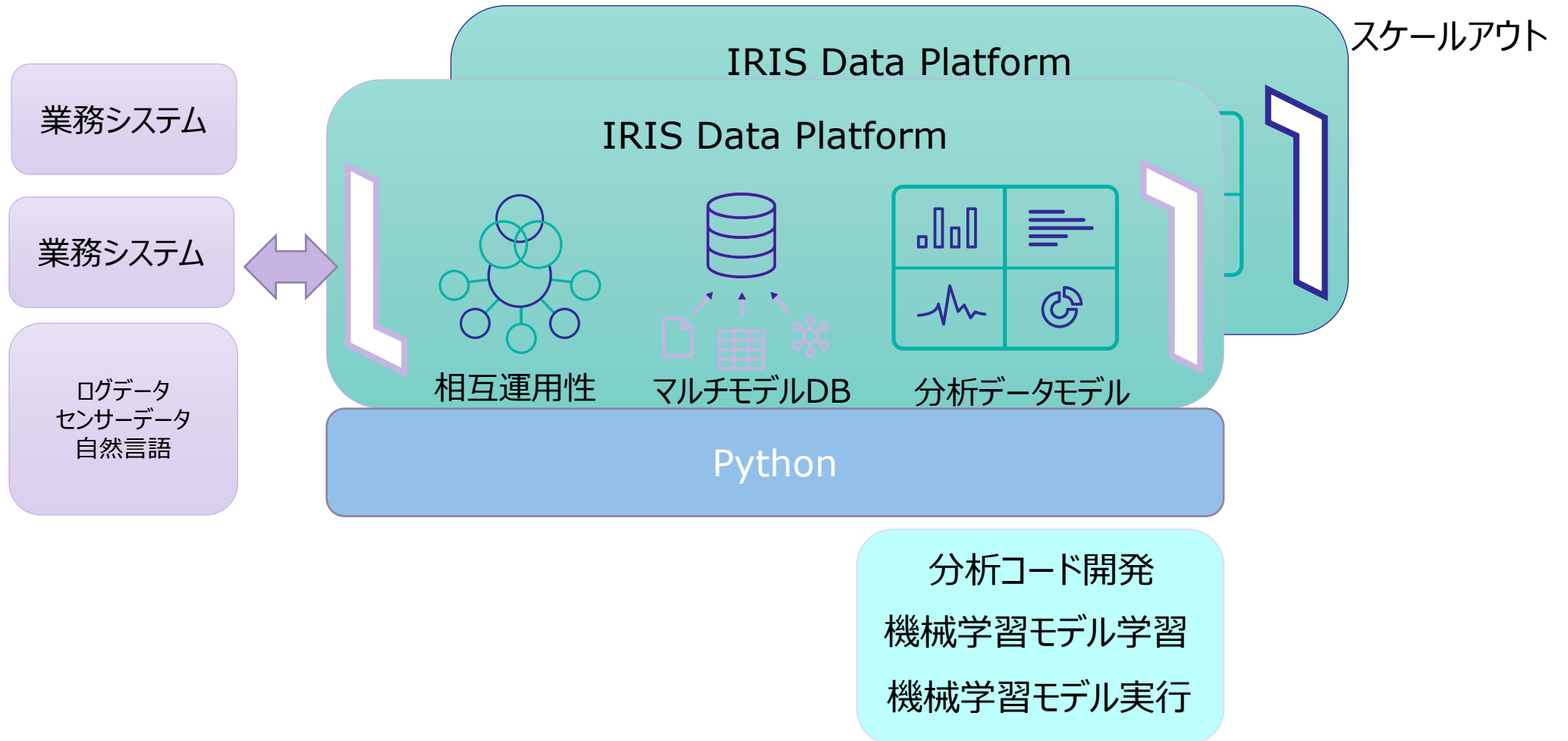
# Embedded Python



- PythonのランタイムをIRISの言語処理モジュールに組み込み
- 膨大なPythonの資産をIRISの「ネイティブ」コードとして利用可能
- PythonプログラマがIRISデータプラットフォームの機能を活用するのが容易
- 技術者の確保が容易



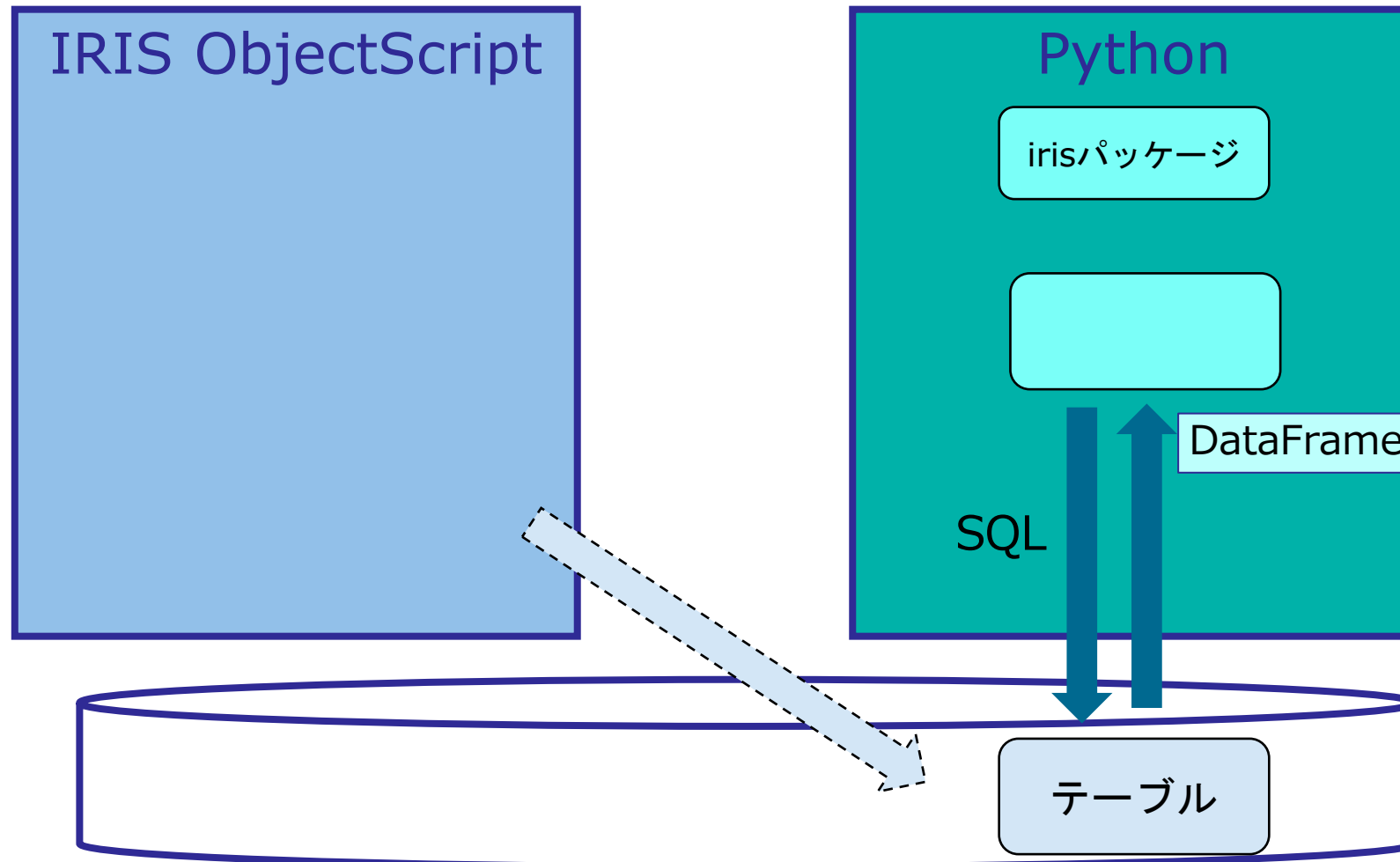
# インターシステムズが提供するプラットフォーム



# Embedded Pythonの機能 (1)



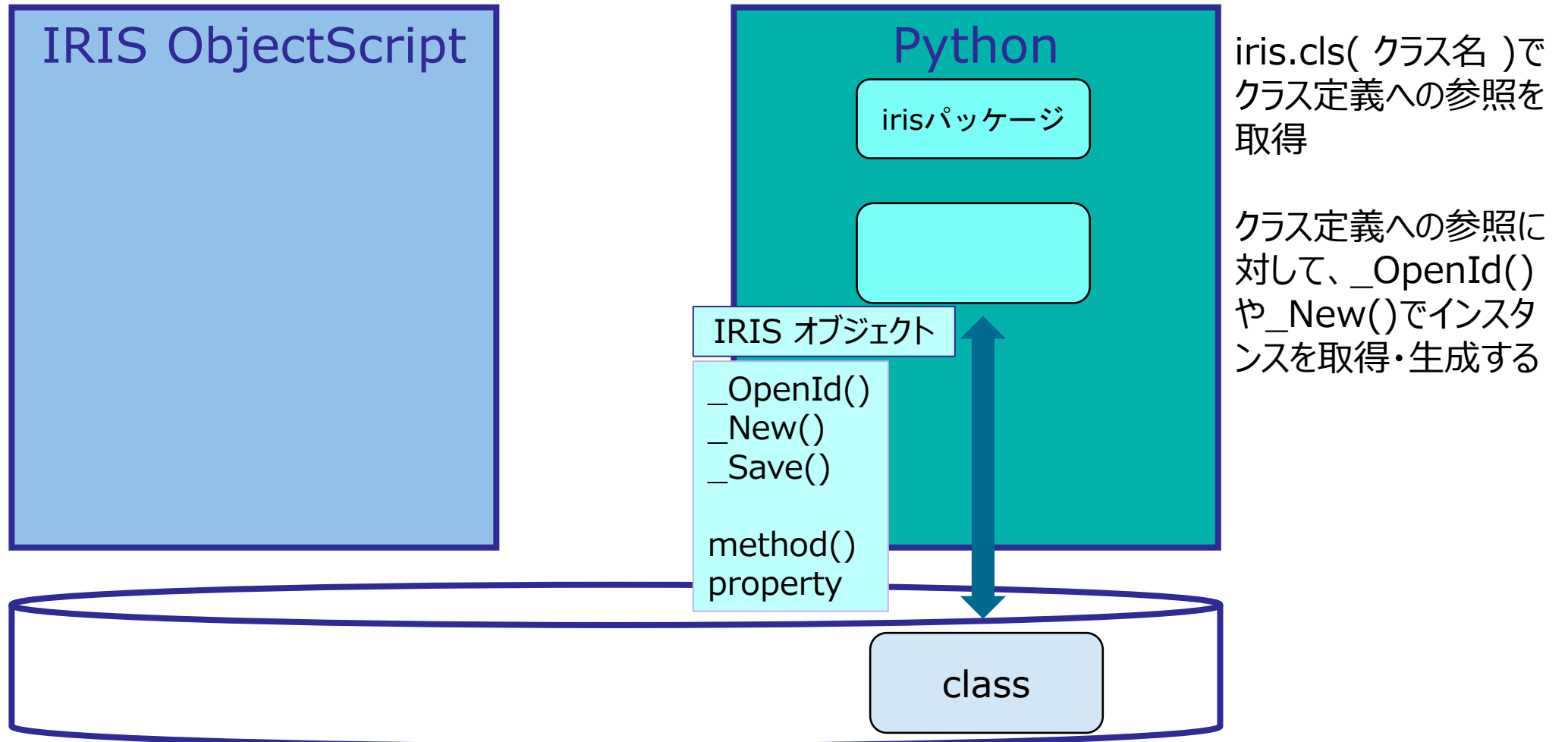
PythonからIRIS上のテーブルにSQLでアクセス



# Embedded Pythonの機能 (2)



PythonからIRISのオブジェクトを操作

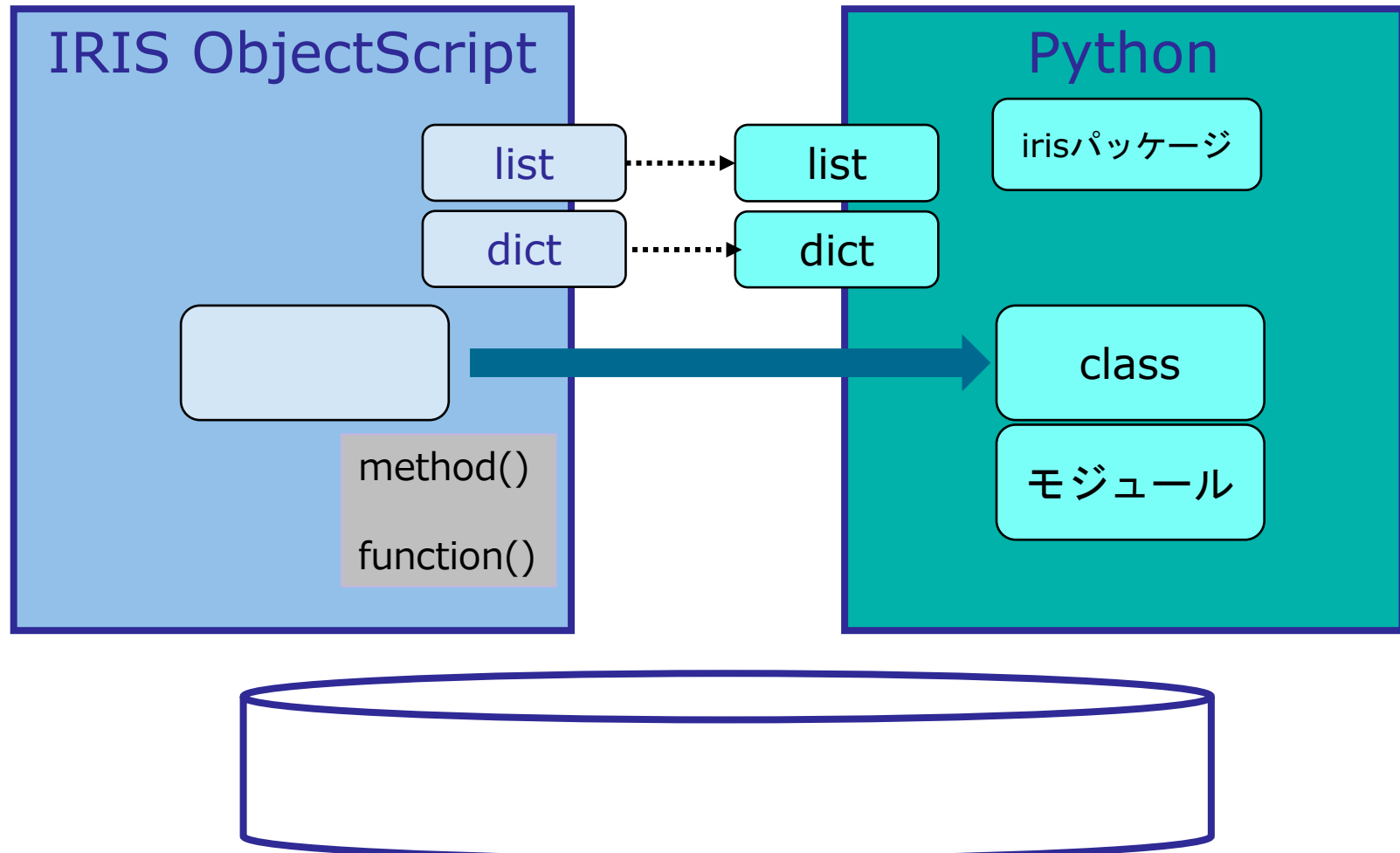


# Embedded Pythonの機能 (3)



ObjectScriptからPythonのクラス・モジュールを操作

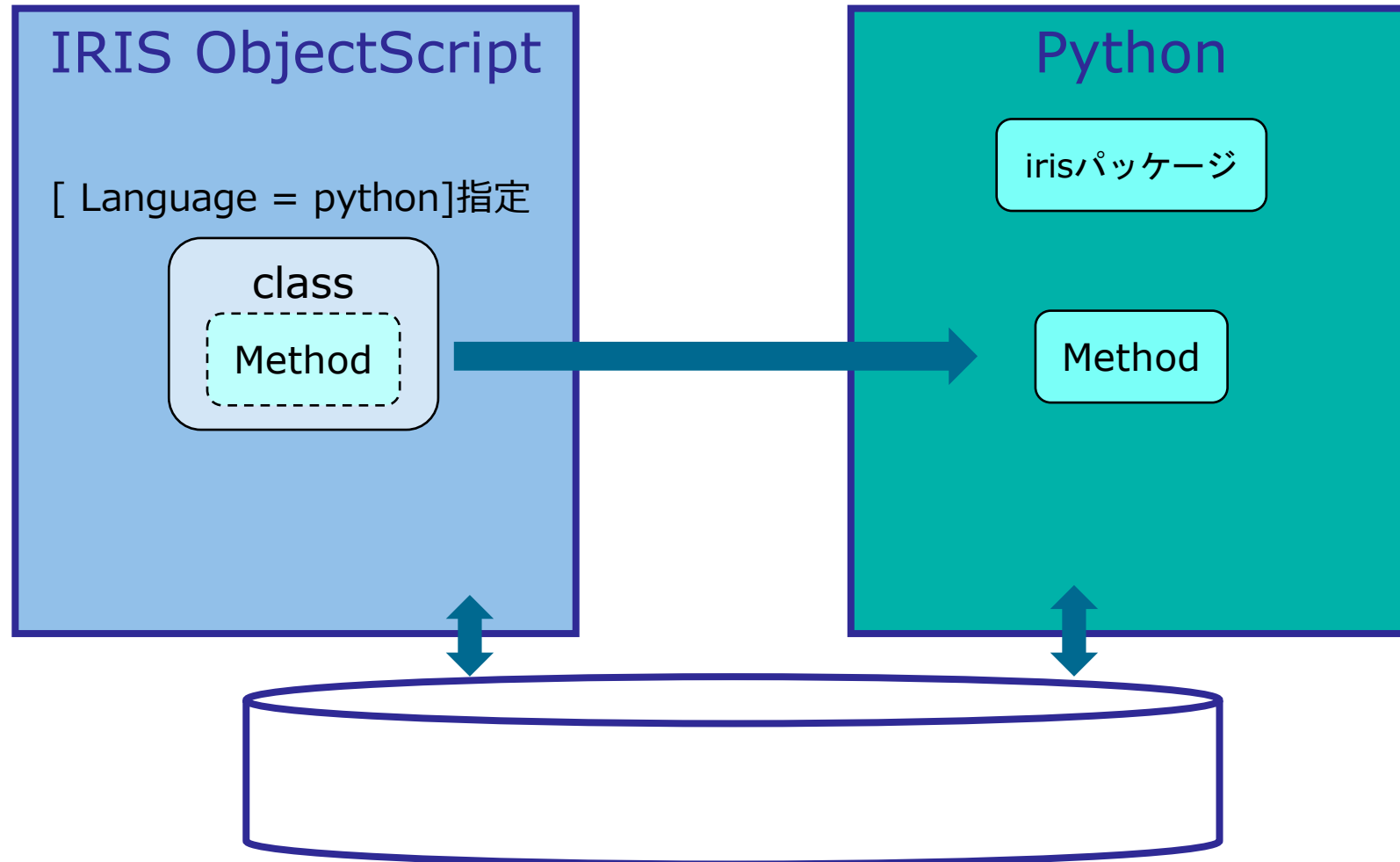
`$system.Python.Import`  
(.pyファイル名)でPython  
モジュールへの参照を取得



# Embedded Pythonの機能 (4)



IRISのクラスのメソッドをPythonで記述



# サンプル



```
SimpleDemo.cls X
src > User > SimpleDemo.cls > User.SimpleDemo > getDistance
26
Debug | Copy Invocation
27 ClassMethod getDistance(origin As %String, destination As %String) As %Integer [ Language = python ]
28 {
29     import googlemaps
30     import iris
31
32     # GoogleMapsのAPIキーをグローバルから読む
33     apikey = iris.gref('^APIKey').get()
34
35     # APIキーを用いてGoogle Mapクライアント生成
36     client = googlemaps.Client(apikey)
37
38     # 2点間の距離を求める
39     r = client.distance_matrix([origin], [destination])
40     distance = int(r['rows'][0]['elements'][0]['distance']['value'])
41
42     return distance
43 }
44
```

問題 3 出力 デバッグ コンソール ターミナル SQL CONSOLE Irisdb + -

```
USER>
USER>
USER>
USER>w #class(SimpleDemo).getDistance("東京駅", "大阪駅")
501002
USER>
```



# サンプル

jupyter simpledemo Last Checkpoint: 2022/02/17 (unsaved changes) ログアウト

ファイル 編集 表示 挿入 セル カーネル ヘルプ Trusted | RISE Python


```
1 7.0 2
2 9.8 5
3 9.8 6
4 9.4 5
...
1138 11.0 6
1139 9.5 6
1140 10.5 5
1141 11.2 6
1142 10.2 5
```

[1143 rows x 12 columns]  
count 1143.000000  
mean 0.086933  
std 0.047267  
min 0.012000  
25% 0.070000  
50% 0.079000  
75% 0.090000  
max 0.611000  
Name: chlorides, dtype: float64

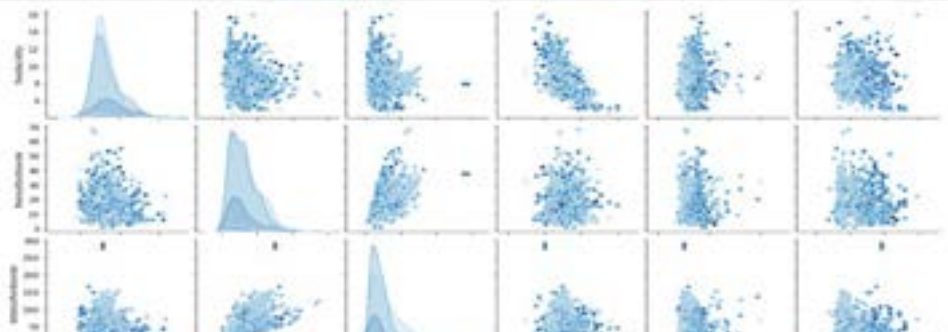
データの性質をさまざまな角度から確かめてみます。Pythonのコードの細かいところは気にしないでください。

In [5]: # qualityフィールドの値のヒストグラムを表示します。  
plt.hist(df.quality, bins=5)

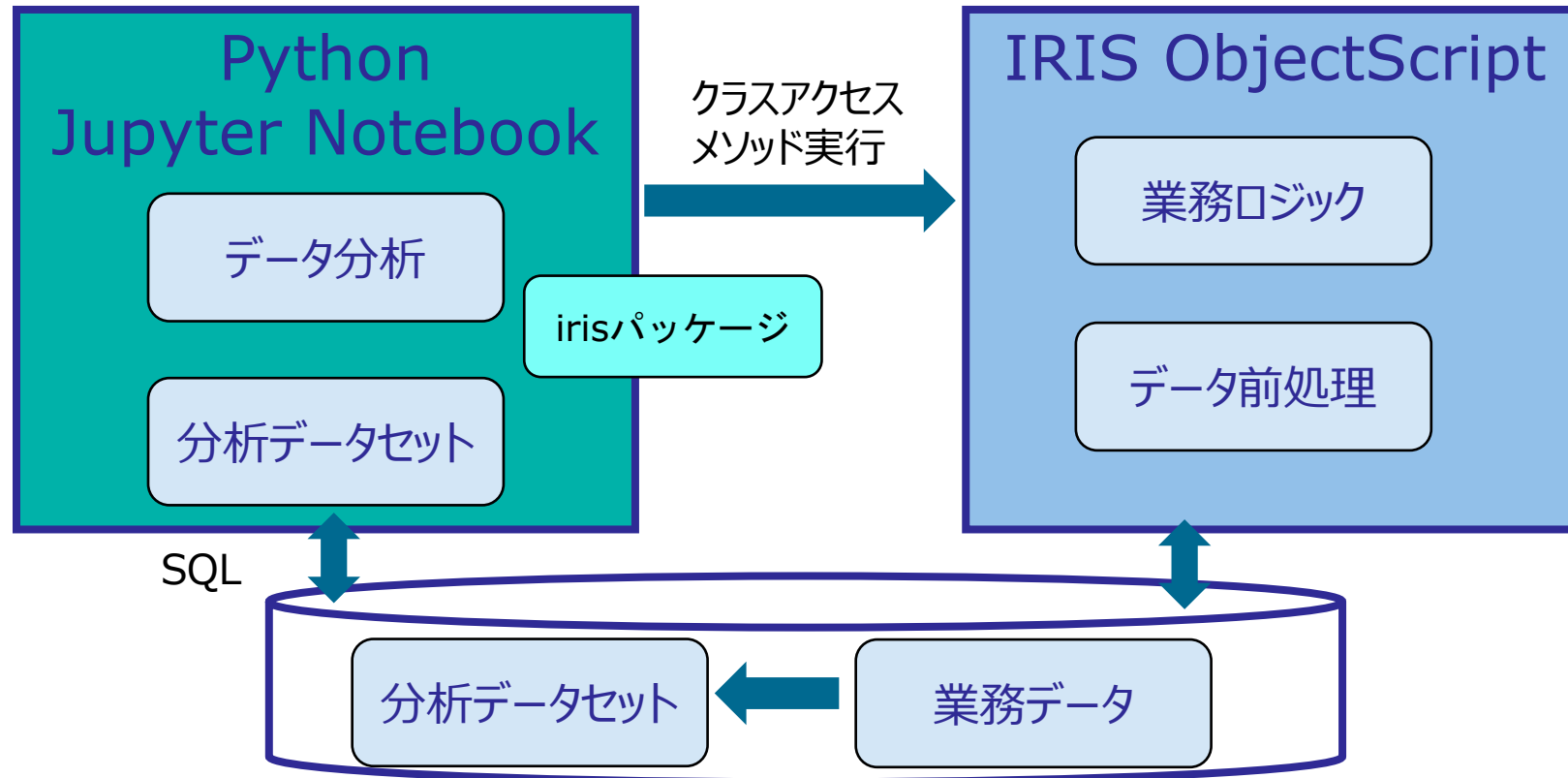
Out [5]: (array([ 6., 33., 483., 462., 159.]),  
array([3., 4., 5., 6., 7., 8.]),  
<BarContainer object of 5 artists>)



In [6]: # 適切なフィールドを選んで散布図行列を作ります。  
# 各点の色でqualityを表します。  
pg = seaborn.pairplot(df[['fixedacidity', 'freesulfurdioxide', 'totalsulfurdioxide', 'sulphates', 'alcohol', 'quality']],  
hue = 'quality', palette = "Blues")



# Jupyter Notebookを使って、IRISからデータを取得して分析



- 最新データに基づいた分析・機械学習が可能
- データ変換や匿名化などの前処理をIRISで記述し、Pythonから呼び出す
- DBA、データエンジニアとデータサイエンティスト、アナリストとの役割分担がスムーズに

# Embedded Python セルフラーニングビデオ公開！

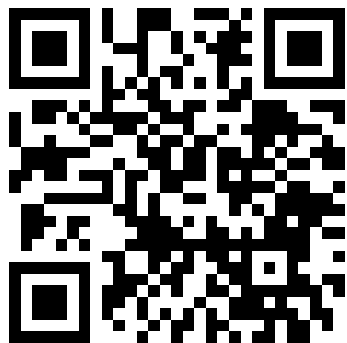


<https://jp.community.intersystems.com/node/520751>

公開中ビデオ (YouTubeプレイリスト) は以下の通りです。

- Embedded Python概要
- 利用前の準備 (ご利用の前に少しでも設定確認が必要です)
- Embedded Pythonでデータベースプログラミング: SQLアクセス編
- Embedded Pythonでデータベースプログラミング: オブジェクトアクセス編
- IRISでPythonを使ってみよう！

その他YouTubeに  
たくさんの動画を公開しています ↓



# インターシステムズ開発者コミュニティ



[jp.community.intersystems.com](https://jp.community.intersystems.com)

- **開発者同士の交流の場**として技術的な質問 & 回答が行えます！
- **ヒントを探す場所**として役立つ記事が見つかります！  
[jp.community.intersystems.com/tags/tips-tricks](https://jp.community.intersystems.com/tags/tips-tricks)
- **学びの場**としてセルフラーニングビデオ公開中！  
[jp.community.intersystems.com/tags/beginner](https://jp.community.intersystems.com/tags/beginner)