Environments

- *Different applications and different scenarios use different ways to interaction with databases.*
- *We use three different connection/interaction models to give students experience with the various options.*

ipython-SQL

In [1]:
```
%load_ext sql
```

In [2]:
```
#
# Set the userid and password for connecting to your instance of SQL.
#
mysql_user = "root"
mysql_password = "dbuserbdbuser"

mysql_url = f"mysql+pymysql://{mysql_user}:{mysql_password}@localhost"

print("Your connection URL is", mysql_url)
```

```
Your connection URL is mysql+pymysql://root:dbuserbdbuser@localhost
```

In [3]:
```
#
# Connect. See the ipython-sql documentation for the $variable syntax.
#
%sql $mysql_url
```

SQL Alchemy and Pandas

In [4]:
```
#
# Yes, I know the cool kids import as pd. I am not cool.
#
import pandas
```

In [5]:
```
#
# Pandas SQL operations require a SQL Alchemy engine.
#
from sqlalchemy import create_engine
```

In [6]:
```
sql_engine = create_engine(mysql_url)
```

pymysql

In [7]:
```
import pymysql
```

In [8]:
```
pymysql_con = pymysql.connect(
    user= mysql_user,
    password= mysql_password,
    host= "localhost",
    port= 3306,
```

```
    autocommit= True,
    cursorclass= pymysql.cursors.DictCursor)
```

Data Loading

Classic Models

- *We will use the Classic Models sample database for many of the questions on this exam.*
- *The directory containing this notebook contains a file* `classic-models-sample.sql`.
- *Load the data:*
  - *Open the file in DataGrip using* `File -> Open` *dialog.*
  - *Select all of the text/SQL in the file.*
  - *Click the green arrowhead to run the files contents.*
- *Running the following queries will test if the load worked.*

In [9]:
```
%sql use classicmodels;
```

```
 * mysql+pymysql://root:***@localhost
0 rows affected.
```

Out[9]:
```
[]
```

In [10]:
```
%sql show tables;
```

```
 * mysql+pymysql://root:***@localhost
8 rows affected.
```

Out[10]:

Tables_in_classicmodels
customers
employees
offices
orderdetails
orders
payments
productlines
products

In [11]:
```
%sql select count(*) as count from orders join orderdetails using(orderNumber)
```

```
 * mysql+pymysql://root:***@localhost
1 rows affected.
```

Out[11]:

count
2996

Lahman's Baseball Database

- *You previously loaded information from Lahman's Baseball Database.*
- *If you have not done so, the following code will load the data into a new schema* `lahmansdb_midterm`.

In [12]:
```
%sql create schema lahmansdb_midterm
```

```
 * mysql+pymysql://root:***@localhost
(pymysql.err.ProgrammingError) (1007, "Can't create database 'lahmansdb_midterm'; dat
abase exists")
[SQL: create schema lahmansdb_midterm]
(Background on this error at: https://sqlalche.me/e/14/f405)
```

In [13]:
```
people_df = pandas.read_csv("./People.csv")
people_df.to_sql("people", schema="lahmansdb_midterm", con=sql_engine,index=False, if_
```

Out[13]: 20370

In [14]:
```
batting_df = pandas.read_csv("./Batting.csv")
batting_df.to_sql("batting", schema="lahmansdb_midterm", con=sql_engine,index=False, 
```

Out[14]: 110495

In [15]:
```
pitching_df = pandas.read_csv("./Pitching.csv")
pitching_df.to_sql("pitching", schema="lahmansdb_midterm", con=sql_engine,index=False,
```

Out[15]: 49430

- *This will test the data loading.*

In [16]:
```
%sql select count(*) as people_count from lahmansdb_midterm.people;
```

```
 * mysql+pymysql://root:***@localhost
1 rows affected.
```

Out[16]:

people_count
20370

In [17]:
```
%sql select count(*) as batting_count from lahmansdb_midterm.batting;
```

```
 * mysql+pymysql://root:***@localhost
1 rows affected.
```

Out[17]:

batting_count
110495

In [18]:
```
%sql select count(*) as pitching_count from lahmansdb_midterm.pitching;
```

```
 * mysql+pymysql://root:***@localhost
1 rows affected.
```