COMS W4111-002 (Fall 2021)
Introduction to Databases
Homework 2: Programming Implement a Simple Database Engine
15 Points

This assignment is due October 22, 11:59 pm EDT

Note: Please replace the information below with your last name, first name and UNI.

Zeng_Xiangyi, UNI:xz2727

Submission

1. File > Print Preview > Save as PDF...
2. Upload .pdf and .ipynb to GradeScope

This assignment is due October 22, 11:59 pm EDT

Collaboration

- You may use any information you get in TA or Prof. Ferguson's office hours, from lectures or from recitations.
- You may use information that you find on the web.
- You are NOT allowed to collaborate with other students outside of office hours.

Part 1: Written & SQL

Written

Please keep your answers brief.

1. Codd's Fourth Rule states that: The data base description is represented at the logical level in the same way as ordinary data, so that authorized users can apply the same relational language to its interrogation as they apply to the regular data. In two sentences please explain this rule and why it is so important.

The database __description__ like __information_schema__ in MySQL is the metadata about all the databases in the DBMS including itself. with Codd's Fourth Rule, we can apply relational language to interrogate the database __description__ in order to get useful information about all the databases in the DBMS, so it is important.

2. Give 3 examples of what would be stored in a database catalog

(1) The "tables" table in the database catalog contains information about all tables of all the databases in the DBMS.
(2) The "columns" table in the database catalog contains information about all the columns of all the tables of all the databases in the DBMS.
(3) The "KEY_COLUMN_USAGE" table in the database catalog contains information about the usage of keys of all tables of all the databases in the DBMS.

3. What is the MySQL database catalog called?

The MySQL database catalog is called **information_schema**.

4. What is the overall goal of indicies in SQL?

The overall goal of indicies in SQL is to make query faster.

5. What are the differences between a primary key and a unique index?

(1) A table can have only one primary key but can have multiple unique indicies.
(2) Primary key can not be NULL, but the unique index can have NULL values.
(3) A clustered index is automatically created by the primary key, but unique index will create non-clustered index by default.

6. Which SELECT statement is more efficient? Why?
- `SELECT playerID,birthState,nameLast,nameFirst FROM people where birthCountry = 'USA' and nameFirst = 'John' and playerID in (select playerID from collegeplaying where schoolID = 'Fordham');`
- `SELECT playerID,birthState,nameLast,nameFirst FROM people NATURAL JOIN collegeplaying where birthCountry = 'USA' and nameFirst = 'John' and schoolID = 'Fordham' group by playerID,birthState,nameLast,nameFirst;`

HINT: SQL uses a query optimizer so you can't just run both of these and see which one performs faster.

For the first one, the execution time is direct proportion to size(collegeplaying)+size(people), and the second one, the execution time is direct proportion to size(people)*size(collegeplaying)+size(people)+group_time. Hence, the first one is more efficient.

7. The create.sql file provided in the zip folder makes a schema and some tables that mimics metadata tables. Note there is the sytax "ON DELETE CASCADE" after the foreign key creation. What does this mean? Why do we want to specify CASCADE for the metadata tables? What does "ON DELETE RESTRICT" mean and when would we generally want to use this?

(1) The syntax "ON DELETE CASCADE" means when the rows of the "parent" table which is referenced by the foreign key of some 'child' table are deleted, the corresponding rows in the "child" table should be deleted as well.

(2) The reason we want to specify CASCADE for the metadata tables is that when some rows in the 'csvtables' table are deleted, we want to cascade this effect to 'csvcolumns' table which references the 'csvtables' by foreign key 'table_name', and then we want to cascade this effect to 'csvindexed' table which references the 'csvcolumns' table by foreign key ('table_name', 'column_name'). By using CASCADE, we can make these metadata tables have data integrity in the relational model.
(3) The syntax "ON DELETE RESTRICT" means that we can not delete a row in 'parent' table when the primary key of this row is referenced by some 'child' table row. If there is no 'child' table row referencing the row in the 'parent' table, we can delete this row in the 'parent' table. we should use this syntax when we don't want 'orphan' rows in the database which can not find the rows in their referencing tables.

SQL

In [2]:
```
%load_ext sql
%sql mysql+pymysql://root:zxy3221915@localhost/lahmansbaseballdb
```

```
The sql extension is already loaded. To reload it, use:
  %reload_ext sql
```

1. Initials

- Find the `initials, firstName, lastName`, for every player from the people table.
- You need to return 10 rows.
- Sort by the nameFirst, nameLast ascending.
- Note: Even for those players with two last names, just return the first letter of their first last name

Answer:

In [31]:
```
%%sql
select concat(ifnull(substr(NameFirst,1,1),'_'),substr(NameLast,1,1))  as initia
        ifnull(NameFirst,'______') as firstName, NameLast as lastName from peopl
        order by firstName, lastname asc
    limit 10;
```

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

Out[31]:

initials	firstName	lastName
_B	______	Boland
_B	______	Booth
_C	______	Carroll
_E	______	Edwards
_E	______	Evans
_F	______	Franklin
_G	______	Gavern

_H	_____	Harrison
_H	_____	Hellings
_H	_____	Higby

The "_" in initials means that the first name is missing. And '_____' in firstName means that the whole first name is missing.

Question 1a): Games Per Player using GROUP BY

- Find the `yearID, lgID, games_per_player`, for every year and league from the appearances table.
- Use a function to round down the games_per_player
- You need to return 10 rows.
- You must use `group by` in this query.

Answer:

In [5]:
```
%%sql
with
  A as (select playerID,yearID,lgID,sum(G_all) as sum_of_games
from appearances
group by playerID, yearID, lgID )
select yearID,lgID,floor(avg(sum_of_games)) as games_per_player from A
group by yearID, lgID limit 10;
```

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

Out[5]:

yearID	lgID	games_per_player
1871	NA	19
1872	NA	22
1873	NA	29
1874	NA	34
1875	NA	33
1876	NL	38
1877	NL	35
1878	NL	43
1879	NL	48
1880	NL	48

First, we should sum the number of games for one player in one year in one league, and then we can compute the average games of player in one year in one league.

Part 2: CSVCatalog Tests

Once you have tested everything successfuly in python, execute your tests one more time in jupyter notebook to show the expected output. You will need to restart your kernel after saving your python files so that jupyter will use the most recent version of your work.

You may need to drop tables before executing your tests one last time so you don't run into integrity errors

In [1]:
```
import unit_test_catalog as cat # This notebook should be in the same directory
```

In [3]:
```
cat.create_table_test()
```

```
Running save core definition
Q = insert into csvtables values(Batting, ./Data/Batting.csv)
Running load core definition
Q = select * from csvtables where table_name = Batting
Running load columns
Q = select * from csvcolumns where table_name = Batting
Running load indexes
Q = select * from csvindexes where table_name = Batting order by index_name,inde
x_order
Table =  {
  "table_name": "Batting",
  "file_name": "./Data/Batting.csv",
  "columns": [],
  "indexes": []
}
Running save core definition
Q = insert into csvtables values(People, ./Data/People.csv)
Running load core definition
Q = select * from csvtables where table_name = People
Running load columns
Q = select * from csvcolumns where table_name = People
Running load indexes
Q = select * from csvindexes where table_name = People order by index_name,index
_order
Table =  {
  "table_name": "People",
  "file_name": "./Data/People.csv",
  "columns": [],
  "indexes": []
}
Running save core definition
Q = insert into csvtables values(Appearances, ./Data/Appearances.csv)
Running load core definition
Q = select * from csvtables where table_name = Appearances
Running load columns
Q = select * from csvcolumns where table_name = Appearances
Running load indexes
Q = select * from csvindexes where table_name = Appearances order by index_name,
index_order
Table =  {
  "table_name": "Appearances",
  "file_name": "./Data/Appearances.csv",
  "columns": [],
  "indexes": []
}
```

In [2]:
```
cat.drop_table_test()# This test would be run after the whole test
```

```
Q = DELETE FROM csvtables WHERE table_name = 'People'
Table 'People' was dropped
Q = DELETE FROM csvtables WHERE table_name = 'Batting'
Table 'Batting' was dropped
Q = DELETE FROM csvtables WHERE table_name = 'Appearances'
Table 'Appearances' was dropped
```

In [4]:
```
cat.add_column_test()
```

```
Running load core definition
Q = select * from csvtables where table_name = Batting
Running load columns
Q = select * from csvcolumns where table_name = Batting
Running load indexes
Q = select * from csvindexes where table_name = Batting order by index_name,inde
x_order
adding column
Q = insert into csvcolumns values(Batting, playerID, text, True)
adding column
Q = insert into csvcolumns values(Batting, yearID, number, True)
adding column
Q = insert into csvcolumns values(Batting, stint, number, True)
Table = {
  "table_name": "Batting",
  "file_name": "./Data/Batting.csv",
  "columns": [
    {
      "column_name": "playerID",
      "column_type": "text",
      "not_null": true
    },
    {
      "column_name": "yearID",
      "column_type": "number",
      "not_null": true
    },
    {
      "column_name": "stint",
      "column_type": "number",
      "not_null": true
    }
  ],
  "indexes": []
}
Running load core definition
Q = select * from csvtables where table_name = People
Running load columns
Q = select * from csvcolumns where table_name = People
Running load indexes
Q = select * from csvindexes where table_name = People order by index_name,index
_order
adding column
Q = insert into csvcolumns values(People, playerID, text, True)
adding column
Q = insert into csvcolumns values(People, birthYear, number, False)
Table = {
  "table_name": "People",
  "file_name": "./Data/People.csv",
```

```
  "columns": [
    {
      "column_name": "playerID",
      "column_type": "text",
      "not_null": true
    },
    {
      "column_name": "birthYear",
      "column_type": "number",
      "not_null": false
    }
  ],
  "indexes": []
}
Running load core definition
Q = select * from csvtables where table_name = Appearances
Running load columns
Q = select * from csvcolumns where table_name = Appearances
Running load indexes
Q = select * from csvindexes where table_name = Appearances order by index_name,
index_order
adding column
Q = insert into csvcolumns values(Appearances, playerID, text, True)
adding column
Q = insert into csvcolumns values(Appearances, yearID, number, True)
adding column
Q = insert into csvcolumns values(Appearances, teamID, number, True)
Table = {
  "table_name": "Appearances",
  "file_name": "./Data/Appearances.csv",
  "columns": [
    {
      "column_name": "playerID",
      "column_type": "text",
      "not_null": true
    },
    {
      "column_name": "yearID",
      "column_type": "number",
      "not_null": true
    },
    {
      "column_name": "teamID",
      "column_type": "number",
      "not_null": true
    }
  ],
  "indexes": []
}
```

In [5]:

```
cat.column_name_failure_test()  # This will throw an error
```

```
Issue!!
---------------------------------------------------------------------------
ValueError                                Traceback (most recent call last)
/tmp/ipykernel_7794/1013433648.py in <module>
----> 1 cat.column_name_failure_test()  # This will throw an error

~/MS_CS/Databases/W4111F21/HomeworkAssignments/HW2/W4111_HW2_Programming/unit_te
st_catalog.py in column_name_failure_test()
```

```
     79 def column_name_failure_test():
     80     cat = CSVCatalog.CSVCatalog()
---> 81     col = CSVCatalog.ColumnDefinition(None, "text", False)
     82     t = cat.get_table("Batting")
     83     t.add_column_definition(col)

~/MS_CS/Databases/W4111F21/HomeworkAssignments/HW2/W4111_HW2_Programming/CSVCata
log.py in __init__(self, column_name, column_type, not_null)
     50         if column_name == None:
     51             print("Issue!!")
---> 52             raise ValueError('You must have a column name!!')
     53         else:
     54             self.column_name = column_name

ValueError: You must have a column name!!
```

In [6]:

```
cat.column_type_failure_test()  # This will throw an error
```

```
Issue!
---------------------------------------------------------------------------
ValueError                                Traceback (most recent call last)
/tmp/ipykernel_7794/2600232317.py in <module>
----> 1 cat.column_type_failure_test()  # This will throw an error

~/MS_CS/Databases/W4111F21/HomeworkAssignments/HW2/W4111_HW2_Programming/unit_te
st_catalog.py in column_type_failure_test()
     94         dbpw="zxy3221915",
     95         db="CSVCatalog")
---> 96     col = CSVCatalog.ColumnDefinition("bird", "canary", False)
     97     t = cat.get_table("Batting")
     98     t.add_column_definition(col)

~/MS_CS/Databases/W4111F21/HomeworkAssignments/HW2/W4111_HW2_Programming/CSVCata
log.py in __init__(self, column_name, column_type, not_null)
     58         else:
     59             print("Issue!")
---> 60             raise ValueError('That column type is not accepted. Please t
ry again.')
     61
     62         if type(not_null)==type(True):

ValueError: That column type is not accepted. Please try again.
```

In [7]:

```
cat.column_not_null_failure_test()  # This will throw an error
```

```
Issue!
---------------------------------------------------------------------------
ValueError                                Traceback (most recent call last)
/tmp/ipykernel_7794/559455694.py in <module>
----> 1 cat.column_not_null_failure_test()  # This will throw an error

~/MS_CS/Databases/W4111F21/HomeworkAssignments/HW2/W4111_HW2_Programming/unit_te
st_catalog.py in column_not_null_failure_test()
    109         dbpw="zxy3221915",
    110         db="CSVCatalog")
--> 111     col = CSVCatalog.ColumnDefinition("name", "text", "happy")
    112     t = cat.get_table("Batting")
    113     t.add_column_definition(col)
```

```
~/MS_CS/Databases/W4111F21/HomeworkAssignments/HW2/W4111_HW2_Programming/CSVCata
log.py in __init__(self, column_name, column_type, not_null)
     64         else:
     65             print("Issue!")
---> 66             raise ValueError('The not_null column must be either True or
False! Please try again.')
     67
     68

ValueError: The not_null column must be either True or False! Please try again.
```

In [8]:

```
cat.add_index_test()
```

```
Running load core definition
Q = select * from csvtables where table_name = Batting
Running load columns
Q = select * from csvcolumns where table_name = Batting
Running load indexes
Q = select * from csvindexes where table_name = Batting order by index_name,inde
x_order
adding index
Q = insert into csvindexes (table_name, column_name, type, index_name, index_ord
er)  values(Batting, playerID, PRIMARY, primary_key, 0)
Q = insert into csvindexes (table_name, column_name, type, index_name, index_ord
er)  values(Batting, yearID, PRIMARY, primary_key, 1)
Q = insert into csvindexes (table_name, column_name, type, index_name, index_ord
er)  values(Batting, stint, PRIMARY, primary_key, 2)
Table = {
  "table_name": "Batting",
  "file_name": "./Data/Batting.csv",
  "columns": [
    {
      "column_name": "playerID",
      "column_type": "text",
      "not_null": true
    },
    {
      "column_name": "stint",
      "column_type": "number",
      "not_null": true
    },
    {
      "column_name": "yearID",
      "column_type": "number",
      "not_null": true
    }
  ],
  "indexes": [
    {
      "index_name": "primary_key",
      "type": "PRIMARY",
      "columns": [
        "playerID",
        "yearID",
        "stint"
      ]
    }
  ]
}
Running load core definition
```

```
Q = select * from csvtables where table_name = People
Running load columns
Q = select * from csvcolumns where table_name = People
Running load indexes
Q = select * from csvindexes where table_name = People order by index_name,index
_order
adding index
Q = insert into csvindexes (table_name, column_name, type, index_name, index_ord
er)  values(People, playerID, PRIMARY, primary_key, 0)
Table = {
  "table_name": "People",
  "file_name": "./Data/People.csv",
  "columns": [
    {
      "column_name": "birthYear",
      "column_type": "number",
      "not_null": false
    },
    {
      "column_name": "playerID",
      "column_type": "text",
      "not_null": true
    }
  ],
  "indexes": [
    {
      "index_name": "primary_key",
      "type": "PRIMARY",
      "columns": [
        "playerID"
      ]
    }
  ]
}
Running load core definition
Q = select * from csvtables where table_name = Appearances
Running load columns
Q = select * from csvcolumns where table_name = Appearances
Running load indexes
Q = select * from csvindexes where table_name = Appearances order by index_name,
index_order
adding index
Q = insert into csvindexes (table_name, column_name, type, index_name, index_ord
er)  values(Appearances, yearID, PRIMARY, primary_key, 0)
Q = insert into csvindexes (table_name, column_name, type, index_name, index_ord
er)  values(Appearances, teamID, PRIMARY, primary_key, 1)
Q = insert into csvindexes (table_name, column_name, type, index_name, index_ord
er)  values(Appearances, playerID, PRIMARY, primary_key, 2)
Table = {
  "table_name": "Appearances",
  "file_name": "./Data/Appearances.csv",
  "columns": [
    {
      "column_name": "playerID",
      "column_type": "text",
      "not_null": true
    },
    {
      "column_name": "teamID",
      "column_type": "number",
      "not_null": true
```

```
    },
    {
      "column_name": "yearID",
      "column_type": "number",
      "not_null": true
    }
  ],
  "indexes": [
    {
      "index_name": "primary_key",
      "type": "PRIMARY",
      "columns": [
        "yearID",
        "teamID",
        "playerID"
      ]
    }
  ]
}
```

In [9]:

```
cat.col_drop_test()
```

```
Running load core definition
Q = select * from csvtables where table_name = Batting
Running load columns
Q = select * from csvcolumns where table_name = Batting
Running load indexes
Q = select * from csvindexes where table_name = Batting order by index_name,inde
x_order
dropping column
Q = delete from csvcolumns where table_name = Batting and column_name  = playerI
D
Column 'playerID' has been dropped!
Running load core definition
Q = select * from csvtables where table_name = Batting
Running load columns
Q = select * from csvcolumns where table_name = Batting
Running load indexes
Q = select * from csvindexes where table_name = Batting order by index_name,inde
x_order
Table = {
  "table_name": "Batting",
  "file_name": "./Data/Batting.csv",
  "columns": [
    {
      "column_name": "stint",
      "column_type": "number",
      "not_null": true
    },
    {
      "column_name": "yearID",
      "column_type": "number",
      "not_null": true
    }
  ],
  "indexes": [
    {
      "index_name": "primary_key",
      "type": "PRIMARY",
      "columns": [
```

```
        "yearID",
        "stint"
      ]
    }
  ]
}
Running load core definition
Q = select * from csvtables where table_name = People
Running load columns
Q = select * from csvcolumns where table_name = People
Running load indexes
Q = select * from csvindexes where table_name = People order by index_name,index
_order
dropping column
Q = delete from csvcolumns where table_name = People and column_name  = birthYea
r
Column 'birthYear' has been dropped!
Running load core definition
Q = select * from csvtables where table_name = People
Running load columns
Q = select * from csvcolumns where table_name = People
Running load indexes
Q = select * from csvindexes where table_name = People order by index_name,index
_order
Table = {
  "table_name": "People",
  "file_name": "./Data/People.csv",
  "columns": [
    {
      "column_name": "playerID",
      "column_type": "text",
      "not_null": true
    }
  ],
  "indexes": [
    {
      "index_name": "primary_key",
      "type": "PRIMARY",
      "columns": [
        "playerID"
      ]
    }
  ]
}
Running load core definition
Q = select * from csvtables where table_name = Appearances
Running load columns
Q = select * from csvcolumns where table_name = Appearances
Running load indexes
Q = select * from csvindexes where table_name = Appearances order by index_name,
index_order
dropping column
Q = delete from csvcolumns where table_name = Appearances and column_name  = tea
mID
Column 'teamID' has been dropped!
Running load core definition
Q = select * from csvtables where table_name = Appearances
Running load columns
Q = select * from csvcolumns where table_name = Appearances
Running load indexes
Q = select * from csvindexes where table_name = Appearances order by index_name,
```

```
index_order
Table = {
  "table_name": "Appearances",
  "file_name": "./Data/Appearances.csv",
  "columns": [
    {
      "column_name": "playerID",
      "column_type": "text",
      "not_null": true
    },
    {
      "column_name": "yearID",
      "column_type": "number",
      "not_null": true
    }
  ],
  "indexes": [
    {
      "index_name": "primary_key",
      "type": "PRIMARY",
      "columns": [
        "yearID",
        "playerID"
      ]
    }
  ]
}
```

In [10]:

```
cat.index_drop_test()
```

```
Running load core definition
Q = select * from csvtables where table_name = Batting
Running load columns
Q = select * from csvcolumns where table_name = Batting
Running load indexes
Q = select * from csvindexes where table_name = Batting order by index_name,inde
x_order
Q = DELETE FROM csvindexes WHERE table_name = 'Batting' and index_name = 'primar
y_key'
Index primary_key has been dropped!
Table = {
  "table_name": "Batting",
  "file_name": "./Data/Batting.csv",
  "columns": [
    {
      "column_name": "stint",
      "column_type": "number",
      "not_null": true
    },
    {
      "column_name": "yearID",
      "column_type": "number",
      "not_null": true
    }
  ],
  "indexes": []
}
Running load core definition
Q = select * from csvtables where table_name = Appearances
Running load columns
```

```
Q = select * from csvcolumns where table_name = Appearances
Running load indexes
Q = select * from csvindexes where table_name = Appearances order by index_name,
index_order
Q = DELETE FROM csvindexes WHERE table_name = 'Appearances' and index_name = 'pr
imary_key'
Index primary_key has been dropped!
Table = {
  "table_name": "Appearances",
  "file_name": "./Data/Appearances.csv",
  "columns": [
    {
      "column_name": "playerID",
      "column_type": "text",
      "not_null": true
    },
    {
      "column_name": "yearID",
      "column_type": "number",
      "not_null": true
    }
  ],
  "indexes": []
}
```

In [11]:

```
cat.describe_table_test()
```

```
Running load core definition
Q = select * from csvtables where table_name = Batting
Running load columns
Q = select * from csvcolumns where table_name = Batting
Running load indexes
Q = select * from csvindexes where table_name = Batting order by index_name,inde
x_order
DESCRIBE Batting =
 {
  "table_name": "Batting",
  "file_name": "./Data/Batting.csv",
  "columns": [
    {
      "column_name": "stint",
      "column_type": "number",
      "not_null": true
    },
    {
      "column_name": "yearID",
      "column_type": "number",
      "not_null": true
    }
  ],
  "indexes": []
}
Running load core definition
Q = select * from csvtables where table_name = people
Running load columns
Q = select * from csvcolumns where table_name = people
Running load indexes
Q = select * from csvindexes where table_name = people order by index_name,index
_order
DESCRIBE People =
```

```
 {
  "table_name": "people",
  "file_name": "./Data/People.csv",
  "columns": [
    {
      "column_name": "playerID",
      "column_type": "text",
      "not_null": true
    }
  ],
  "indexes": [
    {
      "index_name": "primary_key",
      "type": "PRIMARY",
      "columns": [
        "playerID"
      ]
    }
  ]
}
Running load core definition
Q = select * from csvtables where table_name = Appearances
Running load columns
Q = select * from csvcolumns where table_name = Appearances
Running load indexes
Q = select * from csvindexes where table_name = Appearances order by index_name,
index_order
DESCRIBE Appearances =
 {
  "table_name": "Appearances",
  "file_name": "./Data/Appearances.csv",
  "columns": [
    {
      "column_name": "playerID",
      "column_type": "text",
      "not_null": true
    },
    {
      "column_name": "yearID",
      "column_type": "number",
      "not_null": true
    }
  ],
  "indexes": []
}
```

Part 3: CSVTable Tests

In the event that the data sent is too large, jupyter notebook will throw a warning and not print any output. This will happen when you try to retrieve an entire table. Don't worry about getting the output if this happens.

Additonally, the table formatting will get messed up if the columns makes the output too wide. In your tests make sure you project fields so that your outputs are legible.

In [1]:
```
import unit_test_csv_table as tab
```

In [2]:
```
# Drop the tables if you already made them when testing
tab.drop_tables_for_prep()
```

```
Q = DELETE FROM csvtables WHERE table_name = 'people'
Table 'people' was dropped
Q = DELETE FROM csvtables WHERE table_name = 'batting'
Table 'batting' was dropped
Q = DELETE FROM csvtables WHERE table_name = 'appearances'
Table 'appearances' was dropped
```

In [3]:
```
tab.create_lahman_tables()
```

```
Running save core definition
Q = insert into csvtables values(people, ./Data/NewPeople.csv)
Running save core definition
Q = insert into csvtables values(batting, ./Data/NewBatting.csv)
Running save core definition
Q = insert into csvtables values(appearances, ./Data/NewAppearances.csv)
```

In [4]:
```
tab.update_people_columns()
```

```
Running load core definition
Q = select * from csvtables where table_name = people
Running load columns
Q = select * from csvcolumns where table_name = people
Running load indexes
Q = select * from csvindexes where table_name = people order by index_name,index
_order
adding column
Q = insert into csvcolumns values(people, playerID, text, True)
adding column
Q = insert into csvcolumns values(people, birthYear, text, False)
adding column
Q = insert into csvcolumns values(people, birthMonth, text, False)
adding column
Q = insert into csvcolumns values(people, birthDay, text, False)
adding column
Q = insert into csvcolumns values(people, birthCountry, text, False)
adding column
Q = insert into csvcolumns values(people, birthState, text, False)
adding column
Q = insert into csvcolumns values(people, birthCity, text, False)
adding column
Q = insert into csvcolumns values(people, deathYear, text, False)
adding column
Q = insert into csvcolumns values(people, deathMonth, text, False)
adding column
Q = insert into csvcolumns values(people, deathDay, text, False)
adding column
Q = insert into csvcolumns values(people, deathCountry, text, False)
adding column
Q = insert into csvcolumns values(people, deathState, text, False)
adding column
Q = insert into csvcolumns values(people, deathCity, text, False)
adding column
Q = insert into csvcolumns values(people, nameFirst, text, False)
adding column
Q = insert into csvcolumns values(people, nameLast, text, False)
adding column
```

```
Q = insert into csvcolumns values(people, nameGiven, text, False)
adding column
Q = insert into csvcolumns values(people, weight, text, False)
adding column
Q = insert into csvcolumns values(people, height, text, False)
adding column
Q = insert into csvcolumns values(people, bats, text, False)
adding column
Q = insert into csvcolumns values(people, throws, text, False)
adding column
Q = insert into csvcolumns values(people, debut, text, False)
adding column
Q = insert into csvcolumns values(people, finalGame, text, False)
adding column
Q = insert into csvcolumns values(people, retroID, text, False)
adding column
Q = insert into csvcolumns values(people, bbrefID, text, False)
```

In [5]:

```
tab.update_appearances_columns()
```

```
Running load core definition
Q = select * from csvtables where table_name = appearances
Running load columns
Q = select * from csvcolumns where table_name = appearances
Running load indexes
Q = select * from csvindexes where table_name = appearances order by index_name,
index_order
adding column
Q = insert into csvcolumns values(appearances, yearID, text, True)
adding column
Q = insert into csvcolumns values(appearances, teamID, text, True)
adding column
Q = insert into csvcolumns values(appearances, lgID, text, False)
adding column
Q = insert into csvcolumns values(appearances, playerID, text, True)
adding column
Q = insert into csvcolumns values(appearances, G_all, text, False)
adding column
Q = insert into csvcolumns values(appearances, GS, text, False)
adding column
Q = insert into csvcolumns values(appearances, G_batting, text, False)
adding column
Q = insert into csvcolumns values(appearances, G_defense, text, False)
adding column
Q = insert into csvcolumns values(appearances, G_p, text, False)
adding column
Q = insert into csvcolumns values(appearances, G_c, text, False)
adding column
Q = insert into csvcolumns values(appearances, G_1b, text, False)
adding column
Q = insert into csvcolumns values(appearances, G_2b, text, False)
adding column
Q = insert into csvcolumns values(appearances, G_3b, text, False)
adding column
Q = insert into csvcolumns values(appearances, G_ss, text, False)
adding column
Q = insert into csvcolumns values(appearances, G_lf, text, False)
adding column
Q = insert into csvcolumns values(appearances, G_cf, text, False)
adding column
```

```
Q = insert into csvcolumns values(appearances, G_rf, text, False)
adding column
Q = insert into csvcolumns values(appearances, G_of, text, False)
adding column
Q = insert into csvcolumns values(appearances, G_dh, text, False)
adding column
Q = insert into csvcolumns values(appearances, G_ph, text, False)
adding column
Q = insert into csvcolumns values(appearances, G_pr, text, False)
```

In [6]:

```
tab.update_batting_columns()
```

```
Running load core definition
Q = select * from csvtables where table_name = batting
Running load columns
Q = select * from csvcolumns where table_name = batting
Running load indexes
Q = select * from csvindexes where table_name = batting order by index_name,inde
x_order
adding column
Q = insert into csvcolumns values(batting, playerID, text, True)
adding column
Q = insert into csvcolumns values(batting, yearID, text, True)
adding column
Q = insert into csvcolumns values(batting, stint, text, True)
adding column
Q = insert into csvcolumns values(batting, teamID, text, False)
adding column
Q = insert into csvcolumns values(batting, lgID, text, False)
adding column
Q = insert into csvcolumns values(batting, G, text, False)
adding column
Q = insert into csvcolumns values(batting, AB, text, False)
adding column
Q = insert into csvcolumns values(batting, R, text, False)
adding column
Q = insert into csvcolumns values(batting, H, text, False)
adding column
Q = insert into csvcolumns values(batting, 2B, text, False)
adding column
Q = insert into csvcolumns values(batting, 3B, text, False)
adding column
Q = insert into csvcolumns values(batting, HR, text, False)
adding column
Q = insert into csvcolumns values(batting, RBI, text, False)
adding column
Q = insert into csvcolumns values(batting, SB, text, False)
adding column
Q = insert into csvcolumns values(batting, CS, text, False)
adding column
Q = insert into csvcolumns values(batting, BB, text, False)
adding column
Q = insert into csvcolumns values(batting, SO, text, False)
adding column
Q = insert into csvcolumns values(batting, IBB, text, False)
adding column
Q = insert into csvcolumns values(batting, HBP, text, False)
adding column
Q = insert into csvcolumns values(batting, SH, text, False)
adding column
```

```
Q = insert into csvcolumns values(batting, SF, text, False)
adding column
Q = insert into csvcolumns values(batting, GIDP, text, False)
```

In [7]:

```
tab.add_index_definitions()
```

```
Running load core definition
Q = select * from csvtables where table_name = people
Running load columns
Q = select * from csvcolumns where table_name = people
Running load indexes
Q = select * from csvindexes where table_name = people order by index_name,index
_order
adding index
Q = insert into csvindexes (table_name, column_name, type, index_name, index_ord
er)  values(people, playerID, PRIMARY, playerID, 0)
Running load core definition
Q = select * from csvtables where table_name = batting
Running load columns
Q = select * from csvcolumns where table_name = batting
Running load indexes
Q = select * from csvindexes where table_name = batting order by index_name,inde
x_order
adding index
Q = insert into csvindexes (table_name, column_name, type, index_name, index_ord
er)  values(batting, playerID, PRIMARY, playerID, 0)
Q = insert into csvindexes (table_name, column_name, type, index_name, index_ord
er)  values(batting, yearID, PRIMARY, playerID, 1)
Q = insert into csvindexes (table_name, column_name, type, index_name, index_ord
er)  values(batting, stint, PRIMARY, playerID, 2)
Running load core definition
Q = select * from csvtables where table_name = appearances
Running load columns
Q = select * from csvcolumns where table_name = appearances
Running load indexes
Q = select * from csvindexes where table_name = appearances order by index_name,
index_order
adding index
Q = insert into csvindexes (table_name, column_name, type, index_name, index_ord
er)  values(appearances, yearID, PRIMARY, yearID, 0)
Q = insert into csvindexes (table_name, column_name, type, index_name, index_ord
er)  values(appearances, teamID, PRIMARY, yearID, 1)
Q = insert into csvindexes (table_name, column_name, type, index_name, index_ord
er)  values(appearances, playerID, PRIMARY, yearID, 2)
```

In [8]:

```
tab.test_load_info()
```

```
Running load core definition
Q = select * from csvtables where table_name = batting
Running load columns
Q = select * from csvcolumns where table_name = batting
Running load indexes
Q = select * from csvindexes where table_name = batting order by index_name,inde
x_order
./Data/NewBatting.csv
```

In [9]:

```
tab.test_get_col_names()
```

```
Running load core definition
```

```
Q = select * from csvtables where table_name = people
Running load columns
Q = select * from csvcolumns where table_name = people
Running load indexes
Q = select * from csvindexes where table_name = people order by index_name,index
_order
The column names are:  ['bats', 'bbrefID', 'birthCity', 'birthCountry', 'birthDa
y', 'birthMonth', 'birthState', 'birthYear', 'deathCity', 'deathCountry', 'death
Day', 'deathMonth', 'deathState', 'deathYear', 'debut', 'finalGame', 'height',
'nameFirst', 'nameGiven', 'nameLast', 'playerID', 'retroID', 'throws', 'weight']
```

In [10]:

```
tab.add_other_indexes()
```

```
Running load core definition
Q = select * from csvtables where table_name = people
Running load columns
Q = select * from csvcolumns where table_name = people
Running load indexes
Q = select * from csvindexes where table_name = people order by index_name,index
_order
adding index
Q = insert into csvindexes (table_name, column_name, type, index_name, index_ord
er)  values(people, nameLast, INDEX, name, 0)
Q = insert into csvindexes (table_name, column_name, type, index_name, index_ord
er)  values(people, nameFirst, INDEX, name, 1)
Running load core definition
Q = select * from csvtables where table_name = batting
Running load columns
Q = select * from csvcolumns where table_name = batting
Running load indexes
Q = select * from csvindexes where table_name = batting order by index_name,inde
x_order
adding index
Q = insert into csvindexes (table_name, column_name, type, index_name, index_ord
er)  values(batting, teamID, INDEX, teamID, 0)
```

In [11]:

```
# This should throw an error
# Make sure it works properly when you run it in pycharm though!
tab.load_test()
```

```
Running load core definition
Q = select * from csvtables where table_name = people
Running load columns
Q = select * from csvcolumns where table_name = people
Running load indexes
Q = select * from csvindexes where table_name = people order by index_name,index
_order
IOPub data rate exceeded.
The notebook server will temporarily stop sending output
to the client in order to avoid crashing it.
To change this limit, set the config variable
`--NotebookApp.iopub_data_rate_limit`.

Current values:
NotebookApp.iopub_data_rate_limit=1000000.0 (bytes/sec)
NotebookApp.rate_limit_window=3.0 (secs)
```

In [12]:

```
# Might throw an error depending on table size
```

```
# Make sure it works properly when you run it in pycharm though!
tab.dumb_join_test()
```

```
Running load core definition
Q = select * from csvtables where table_name = batting
Running load columns
Q = select * from csvcolumns where table_name = batting
Running load indexes
Q = select * from csvindexes where table_name = batting order by index_name,inde
x_order
Running load core definition
Q = select * from csvtables where table_name = appearances
Running load columns
Q = select * from csvcolumns where table_name = appearances
Running load indexes
Q = select * from csvindexes where table_name = appearances order by index_name,
index_order
Processed 200 left rows.
Processed 400 left rows.
Processed 600 left rows.
Processed 800 left rows.
Processed 1000 left rows.
Processed 1200 left rows.
Processed 1400 left rows.
Processed 1600 left rows.
Processed 1800 left rows.
Processed 2000 left rows.
Processed 2200 left rows.
Processed 2400 left rows.
Processed 2600 left rows.
Processed 2800 left rows.
Processed 3000 left rows.
Processed 3200 left rows.
Processed 3400 left rows.
Processed 3600 left rows.
Processed 3800 left rows.
Processed 4000 left rows.
Processed 4200 left rows.
Processed 4400 left rows.
Processed 4600 left rows.
Processed 4800 left rows.
Processed 5000 left rows.
+------------+----------+----------+------+-----+---------+-------------+
| playerID   |   yearID | teamID   |   AB |   H |   G_all |   G_batting |
+============+==========+==========+======+=====+=========+=============+
| baxtemi01  |     2010 | SDN      |    8 |   1 |       9 |           9 |
+------------+----------+----------+------+-----+---------+-------------+
| baxtemi01  |     2011 | NYN      |   34 |   8 |      22 |          22 |
+------------+----------+----------+------+-----+---------+-------------+
| baxtemi01  |     2012 | NYN      |  179 |  47 |      89 |          89 |
+------------+----------+----------+------+-----+---------+-------------+
| baxtemi01  |     2013 | NYN      |  132 |  25 |      74 |          74 |
+------------+----------+----------+------+-----+---------+-------------+
| baxtemi01  |     2014 | LAN      |    7 |   0 |       4 |           4 |
+------------+----------+----------+------+-----+---------+-------------+
| baxtemi01  |     2015 | CHN      |   57 |  14 |      34 |          34 |
+------------+----------+----------+------+-----+---------+-------------+
```

In [14]:

```
tab.get_access_path_test()
```

```
Running load core definition
Q = select * from csvtables where table_name = batting
Running load columns
Q = select * from csvcolumns where table_name = batting
Running load indexes
Q = select * from csvindexes where table_name = batting order by index_name,inde
x_order
teamID
124
```

In [15]:
```
tab.sub_where_template_test()
```

```
Running load core definition
Q = select * from csvtables where table_name = people
Running load columns
Q = select * from csvcolumns where table_name = people
Running load indexes
Q = select * from csvindexes where table_name = people order by index_name,index
_order
The where_template is : {"nameFirst": "David","teamID":"CHA"}
The sub_where_template is: {'nameFirst': 'David'}
```

In [16]:
```
tab.test_find_by_template_index()
```

```
Running load core definition
Q = select * from csvtables where table_name = batting
Running load columns
Q = select * from csvcolumns where table_name = batting
Running load indexes
Q = select * from csvindexes where table_name = batting order by index_name,inde
x_order
find_by_template_index of template:{"yearID":"1965","teamID":"ML1"}, index: "tea
mID"
[{'playerID': 'aaronha01', 'yearID': '1965', 'teamID': 'ML1', 'lgID': 'NL'}, {'p
layerID': 'aaronto01', 'yearID': '1965', 'teamID': 'ML1', 'lgID': 'NL'}, {'playe
rID': 'alomasa01', 'yearID': '1965', 'teamID': 'ML1', 'lgID': 'NL'}, {'playerI
D': 'aloufe01', 'yearID': '1965', 'teamID': 'ML1', 'lgID': 'NL'}]
```

In [17]:
```
tab.smart_join_test()
```

```
Running load core definition
Q = select * from csvtables where table_name = batting
Running load columns
Q = select * from csvcolumns where table_name = batting
Running load indexes
Q = select * from csvindexes where table_name = batting order by index_name,inde
x_order
Running load core definition
Q = select * from csvtables where table_name = appearances
Running load columns
Q = select * from csvcolumns where table_name = appearances
Running load indexes
Q = select * from csvindexes where table_name = appearances order by index_name,
index_order
Processed 2 left rows.
Processed 4 left rows.
Processed 6 left rows.
+------------+----------+----------+------+-----+---------+-------------+
| playerID   |   yearID | teamID   |   AB |   H |   G_all |   G_batting |
```

```
+============+==========+==========+======+=====+=========+=============+
| baxtemi01  |     2010 | SDN      |    8 |   1 |       9 |           9 |
+------------+----------+----------+------+-----+---------+-------------+
| baxtemi01  |     2011 | NYN      |   34 |   8 |      22 |          22 |
+------------+----------+----------+------+-----+---------+-------------+
| baxtemi01  |     2012 | NYN      |  179 |  47 |      89 |          89 |
+------------+----------+----------+------+-----+---------+-------------+
| baxtemi01  |     2013 | NYN      |  132 |  25 |      74 |          74 |
+------------+----------+----------+------+-----+---------+-------------+
| baxtemi01  |     2014 | LAN      |    7 |   0 |       4 |           4 |
+------------+----------+----------+------+-----+---------+-------------+
| baxtemi01  |     2015 | CHN      |   57 |  14 |      34 |          34 |
+------------+----------+----------+------+-----+---------+-------------+
```

In [18]:

```
# Compare the time it takes to do the dumb join and the smart join below
#This is a timer that will track how long it takes to execute your cell.
# Times will vary based on how long it takes to query your AWS Server, but you s

#----Your Code Here----
```

In [22]:

```
%%time
tab.dumb_join_test()
```

```
Running load core definition
Q = select * from csvtables where table_name = batting
Running load columns
Q = select * from csvcolumns where table_name = batting
Running load indexes
Q = select * from csvindexes where table_name = batting order by index_name,inde
x_order
Running load core definition
Q = select * from csvtables where table_name = appearances
Running load columns
Q = select * from csvcolumns where table_name = appearances
Running load indexes
Q = select * from csvindexes where table_name = appearances order by index_name,
index_order
Processed 200 left rows.
Processed 400 left rows.
Processed 600 left rows.
Processed 800 left rows.
Processed 1000 left rows.
Processed 1200 left rows.
Processed 1400 left rows.
Processed 1600 left rows.
Processed 1800 left rows.
Processed 2000 left rows.
Processed 2200 left rows.
Processed 2400 left rows.
Processed 2600 left rows.
Processed 2800 left rows.
Processed 3000 left rows.
Processed 3200 left rows.
Processed 3400 left rows.
Processed 3600 left rows.
Processed 3800 left rows.
Processed 4000 left rows.
Processed 4200 left rows.
Processed 4400 left rows.
```

```
Processed 4600 left rows.
Processed 4800 left rows.
Processed 5000 left rows.
+------------+----------+----------+------+-----+---------+-------------+
| playerID   |   yearID | teamID   |   AB |   H |   G_all |   G_batting |
+============+==========+==========+======+=====+=========+=============+
| baxtemi01  |     2010 | SDN      |    8 |   1 |       9 |           9 |
+------------+----------+----------+------+-----+---------+-------------+
| baxtemi01  |     2011 | NYN      |   34 |   8 |      22 |          22 |
+------------+----------+----------+------+-----+---------+-------------+
| baxtemi01  |     2012 | NYN      |  179 |  47 |      89 |          89 |
+------------+----------+----------+------+-----+---------+-------------+
| baxtemi01  |     2013 | NYN      |  132 |  25 |      74 |          74 |
+------------+----------+----------+------+-----+---------+-------------+
| baxtemi01  |     2014 | LAN      |    7 |   0 |       4 |           4 |
+------------+----------+----------+------+-----+---------+-------------+
| baxtemi01  |     2015 | CHN      |   57 |  14 |      34 |          34 |
+------------+----------+----------+------+-----+---------+-------------+
CPU times: user 14.9 s, sys: 52.2 ms, total: 15 s
Wall time: 15.1 s
```

In [21]:

```
%%time
tab.smart_join_test()
```

```
Running load core definition
Q = select * from csvtables where table_name = batting
Running load columns
Q = select * from csvcolumns where table_name = batting
Running load indexes
Q = select * from csvindexes where table_name = batting order by index_name,inde
x_order
Running load core definition
Q = select * from csvtables where table_name = appearances
Running load columns
Q = select * from csvcolumns where table_name = appearances
Running load indexes
Q = select * from csvindexes where table_name = appearances order by index_name,
index_order
Processed 2 left rows.
Processed 4 left rows.
Processed 6 left rows.
+------------+----------+----------+------+-----+---------+-------------+
| playerID   |   yearID | teamID   |   AB |   H |   G_all |   G_batting |
+============+==========+==========+======+=====+=========+=============+
| baxtemi01  |     2010 | SDN      |    8 |   1 |       9 |           9 |
+------------+----------+----------+------+-----+---------+-------------+
| baxtemi01  |     2011 | NYN      |   34 |   8 |      22 |          22 |
+------------+----------+----------+------+-----+---------+-------------+
| baxtemi01  |     2012 | NYN      |  179 |  47 |      89 |          89 |
+------------+----------+----------+------+-----+---------+-------------+
| baxtemi01  |     2013 | NYN      |  132 |  25 |      74 |          74 |
+------------+----------+----------+------+-----+---------+-------------+
| baxtemi01  |     2014 | LAN      |    7 |   0 |       4 |           4 |
+------------+----------+----------+------+-----+---------+-------------+
| baxtemi01  |     2015 | CHN      |   57 |  14 |      34 |          34 |
+------------+----------+----------+------+-----+---------+-------------+
CPU times: user 652 ms, sys: 3 µs, total: 652 ms
Wall time: 661 ms
```