Part 3:

```
use performance_schema;
insert into setup_objects
values('EVENT','Lahman2016','%','YES', 'YES');

select nameFirst, nameLast, max(RBI) from Batting inner join
Master using (playerID) where HR = 0 limit 1;
select (timer_end - TIMER_START)/1000000 as Execution_Time_us
from performance_schema.events_transactions_current order by
THREAD_ID desc limit 1;
```

Firstly, check the execution time without indexing:

	Execution Time (us)
Trial_1	96196758
Trial_2	93663081
Trial_3	95176147
Trial_4	94436145
Trial_5	105501087
Avg Time	96994643.6

The preformance is pretty bad.
Then, use “Explain” to examine the query:

```
Explain select nameFirst, nameLast, max(RBI) from Batting
inner join Master using (playerID) where HR = 0 limit 1;
```

```
+----+-------------+--------+------------+------+---------------+------+---------+------+--------+----------+----------------------------------------------------+
| id | select_type | table  | partitions | type | possible_keys | key  | key_len | ref  | rows   | filtered | Extra                                              |
+----+-------------+--------+------------+------+---------------+------+---------+------+--------+----------+----------------------------------------------------+
|  1 | SIMPLE      | Batting | NULL       | ALL  | NULL          | NULL | NULL    | NULL | 102262 |    10.00 | Using where                                        |
|  1 | SIMPLE      | Master  | NULL       | ALL  | NULL          | NULL | NULL    | NULL |  18901 |    10.00 | Using where; Using join buffer (Block Nested Loop) |
+----+-------------+--------+------------+------+---------------+------+---------+------+--------+----------+----------------------------------------------------+
2 rows in set, 1 warning (0.00 sec)
```

It is searching without index. First thing we should try is to add the primary keys for both Master and Batting table and try again.

```
ALTER TABLE Master add PRIMARY KEY (playerID);
ALTER TABLE Batting ADD CONSTRAINT PK_Batting PRIMARY KEY
(playerID,yearID,stint);
```

Secondly, check the execution time with primary keys on playerID:

	Execution Time (us)
Trial_1	70956
Trial_2	68702
Trial_3	74392
Trial_4	69001
Trial_5	70711
Avg Time	70752.4

The performance improved enormously. Use “Explain” to examine the query again,

```
Explain select nameFirst, nameLast, max(RBI) from Batting
inner join Master using (playerID) where HR = 0 limit 1;
```

```
+----+-------------+---------+------------+--------+---------------+---------+---------+----------------------------+--------+----------+-------------+
| id | select_type | table   | partitions | type   | possible_keys | key     | key_len | ref                        | rows   | filtered | Extra       |
+----+-------------+---------+------------+--------+---------------+---------+---------+----------------------------+--------+----------+-------------+
|  1 | SIMPLE      | Batting | NULL       | ALL    | PRIMARY       | NULL    | NULL    | NULL                       | 102316 |    10.00 | Using where |
|  1 | SIMPLE      | Master  | NULL       | eq_ref | PRIMARY       | PRIMARY | 767     | lahman2016.Batting.playerID |      1 |   100.00 | NULL        |
+----+-------------+---------+------------+--------+---------------+---------+---------+----------------------------+--------+----------+-------------+
2 rows in set, 1 warning (0.01 sec)
```

We can see that the query is already using the index from the primary key of “PlayerID” from Master. It seems that Batting is using a keyword “where. So maybe adding an index for HR in Batting table might help with the performance.

```
CREATE INDEX index_Master_nameFirst ON Master(nameFirst) USING
BTREE;
```

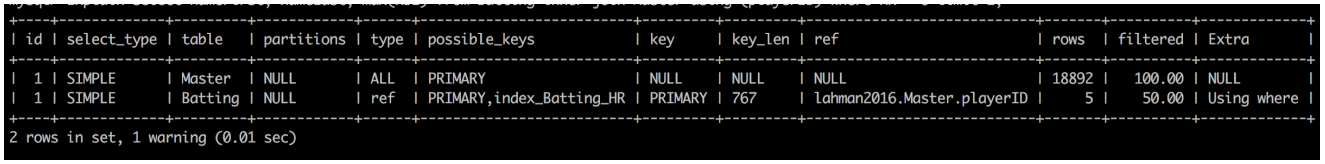
```
select nameFirst, nameLast, max(RBI) from Batting inner join
Master using (playerID) where HR = 0 limit 1;
select (timer_end - TIMER_START)/1000000 as Execution_Time_us
from performance_schema.events_transactions_current order by
THREAD_ID desc limit 1;
```

Execution time with primary keys on playerID and index on Batting.HR:

	Execution Time (us)
Trial_1	107522
Trial_2	103096
Trial_3	111755
Trial_4	109873
Trial_5	107246
Avg Time	107898.4

The performance actually become worse.

```
Explain select nameFirst, nameLast, max(RBI) from Batting
inner join Master using (playerID) where HR = 0 limit 1;
```



```
+----+-------------+--------+------------+------+-------------------------+---------+---------+---------------------------+-------+----------+-------------+
| id | select_type | table  | partitions | type | possible_keys           | key     | key_len | ref                       | rows  | filtered | Extra       |
+----+-------------+--------+------------+------+-------------------------+---------+---------+---------------------------+-------+----------+-------------+
|  1 | SIMPLE      | Master | NULL       | ALL  | PRIMARY                 | NULL    | NULL    | NULL                      | 18892 |   100.00 | NULL        |
|  1 | SIMPLE      | Batting| NULL       | ref  | PRIMARY,index_Batting_HR| PRIMARY | 767     | lahman2016.Master.playerID|     5 |    50.00 | Using where |
+----+-------------+--------+------------+------+-------------------------+---------+---------+---------------------------+-------+----------+-------------+
2 rows in set, 1 warning (0.01 sec)
```

The query is actually using the HR index instead of the playerID primary key. HR might have the same number among the rows, so it is not an unique index. But primary key must be an unique index without null value. This is the reason why the performance become worse. So remove the newly added HR index

```
ALTER TABLE Batting DROP INDEX index_Batting_HR;
```

Thirdly, By adding a foreign key Batting.playerID => Master.playerID might help with the inner join:

```
ALTER TABLE Batting ADD CONSTRAINT fk_Batting_Master
FOREIGN KEY (playerID) REFERENCES Master(playerID);
```

```
select nameFirst, nameLast, max(RBI) from Batting inner join
Master using (playerID) where HR = 0 limit 1;
```

```
select (timer_end - TIMER_START)/1000000 as Execution_Time_us
from performance_schema.events_transactions_current order by
THREAD_ID desc limit 1;
```

Execution time with primary keys on playerID and Foreign key on Batting.playerID => Master.playerID:

	Execution Time (us)
Trial_1	123630
Trial_2	118224
Trial_3	115251
Trial_4	111590
Trial_5	113186
Avg Time	116376.2

```
Explain select nameFirst, nameLast, max(RBI) from Batting
inner join Master using (playerID) where HR = 0 limit 1;
```

```
+----+-------------+---------+------------+------+-------------------------------+---------+---------+---------------------------+-------+----------+-------------+
| id | select_type | table   | partitions | type | possible_keys                 | key     | key_len | ref                       | rows  | filtered | Extra       |
+----+-------------+---------+------------+------+-------------------------------+---------+---------+---------------------------+-------+----------+-------------+
|  1 | SIMPLE      | Master  | NULL       | ALL  | PRIMARY                       | NULL    | NULL    | NULL                      | 18892 |   100.00 | NULL        |
|  1 | SIMPLE      | Batting | NULL       | ref  | PRIMARY,index_Batting_nameLast | PRIMARY | 767     | lahman2016.Master.playerID |     1 |    10.00 | Using where |
+----+-------------+---------+------------+------+-------------------------------+---------+---------+---------------------------+-------+----------+-------------+
2 rows in set, 1 warning (0.00 sec)
```

The performance becomes even worse. The query is using the Batting.playerID foreign key instead of the Master.playerID primary key which will again decrease the speed. So remove the newly added foreign key.

```
ALTER TABLE Batting DROP FOREIGN KEY fk_Batting_Master;
```

Fourthly, add an index for either nameFirst or nameLast in Master with primary keys on playerID for testing purpose (since adding those index shouldn't improve the performance thoericatially):

```
CREATE INDEX index_Master_nameFirst ON Master(nameFirst) USING
BTREE;

select nameFirst, nameLast, max(RBI) from Batting inner join
Master using (playerID) where HR = 0 limit 1;
select (timer_end - TIMER_START)/1000000 as Execution_Time_us
from performance_schema.events_transactions_current order by
THREAD_ID desc limit 1;
```

For primary keys on playerID and nameFirst index,

	Execution Time (us)
Trial_1	73038
Trial_2	74697
Trial_3	70520
Trial_4	70560
Trial_5	74090
Avg Time	72581

```
ALTER TABLE Master DROP INDEX index_Master_nameFirst;

CREATE INDEX index_Master_nameLast ON Master(nameLast) USING
BTREE;
select nameFirst, nameLast, max(RBI) from Batting inner join
Master using (playerID) where HR = 0 limit 1;
select (timer_end - TIMER_START)/1000000 as Execution_Time_us
from performance_schema.events_transactions_current order by
THREAD_ID desc limit 1;
```

For primary keys on playerID and nameLast index,

	Execution Time (us)
Trial_1	69397
Trial_2	70993
Trial_3	72346
Trial_4	73799
Trial_5	73442
Avg Time	71995.4

```
ALTER TABLE Master DROP INDEX index_Master_nameLast;

CREATE INDEX index_Master_nameFirst ON Master(nameFirst) USING
BTREE;
CREATE INDEX index_Master_nameLast ON Master(nameLast) USING
BTREE;

select nameFirst, nameLast, max(RBI) from Batting inner join
Master using (playerID) where HR = 0 limit 1;
select (timer_end - TIMER_START)/1000000 as Execution_Time_us
from performance_schema.events_transactions_current order by
THREAD_ID desc limit 1;
```

For primary keys on playerID and both index on nameFirst and nameLast,

	Execution Time (us)
Trial_1	72560
Trial_2	74342
Trial_3	75063
Trial_4	70840
Trial_5	70287
Avg Time	72618.4

The time increased by a small amount. It proves our guess that it is useless to add index on nameFirst or nameLast.

```
ALTER TABLE Master DROP INDEX index_Master_nameFirst;
ALTER TABLE Master DROP INDEX index_Master_nameLast;
```

Because of aggregation, adding index on Batting.RBI is not useful.

```
CREATE INDEX index_Batting_RBI ON Batting(RBI) USING BTREE;

select nameFirst, nameLast, max(RBI) from Batting inner join
Master using (playerID) where HR = 0 limit 1;
select (timer_end - TIMER_START)/1000000 as Execution_Time_us
from performance_schema.events_transactions_current order by
THREAD_ID desc limit 1;
```

For primary keys on playerID and RBI index,

	Execution Time (us)
Trial_1	73566
Trial_2	71987
Trial_3	73153
Trial_4	71496
Trial_5	70633
Avg Time	72167

This proves that adding index on Batting.RBI is not useful.

```
ALTER TABLE Batting DROP INDEX index_Batting_RBI;
```

All other combination is meaningless. So the best way to improve the performance is to add primary keys for both Master.playerID and Batting.playerID only.