First Advanced Query

SELECT r.name, AVG(d.Price) AS AvgPrice
FROM Restaurant r JOIN Dishes d ON (r.restaurantID = d.RestaurantID)
GROUP BY r.restaurantID;

<u>Default Index Designs</u>

The following indexing snippets show the default indexes generated automatically by the database engine for all the relations (Restaurant, Dishes, Purchases and Reviews) using their defined PRIMARY and FOREIGN key constraints.

```
mysql> SHOW INDEX FROM Restaurant;
+------------+------------+----------+--------------+--------------+-----------+-------------+----------+--------+------+------------+---------+---------------+---------+------------+
| Table      | Non_unique | Key_name | Seq_in_index | Column_name  | Collation | Cardinality | Sub_part | Packed | Null | Index_type | Comment |
Index_comment | Visible | Expression |
+------------+------------+----------+--------------+--------------+-----------+-------------+----------+--------+------+------------+---------+---------------+---------+------------+
| Restaurant |          0 | PRIMARY  |            1 | restaurantID | A         |        1136 |     NULL |   NULL |      | BTREE      |         |               | YES     | NULL       |
+------------+------------+----------+--------------+--------------+-----------+-------------+----------+--------+------+------------+---------+---------------+---------+------------+
1 row in set (0.01 sec)

mysql> SHOW INDEX FROM Dishes;
+--------+------------+--------------+--------------+--------------+-----------+-------------+----------+--------+------+------------+---------+---------------+---------+------------+
| Table  | Non_unique | Key_name     | Seq_in_index | Column_name  | Collation | Cardinality | Sub_part | Packed | Null | Index_type | Comment |
Index_comment | Visible | Expression |
+--------+------------+--------------+--------------+--------------+-----------+-------------+----------+--------+------+------------+---------+---------------+---------+------------+
| Dishes |          0 | PRIMARY      |            1 | Name         | A         |        1000 |     NULL |   NULL |      | BTREE      |         |               | YES     | NULL       |
| Dishes |          1 | RestaurantID |            1 | RestaurantID | A         |         616 |     NULL |   NULL |      | BTREE      |         |               | YES     | NULL       |
+--------+------------+--------------+--------------+--------------+-----------+-------------+----------+--------+------+------------+---------+---------------+---------+------------+
2 rows in set (0.01 sec)

mysql> SHOW INDEX FROM Purchases;
+-----------+------------+--------------+--------------+--------------+-----------+-------------+----------+--------+------+------------+---------+---------------+---------+------------+
| Table     | Non_unique | Key_name     | Seq_in_index | Column_name  | Collation | Cardinality | Sub_part | Packed | Null | Index_type | Comment |
Index_comment | Visible | Expression |
+-----------+------------+--------------+--------------+--------------+-----------+-------------+----------+--------+------+------------+---------+---------------+---------+------------+
| Purchases |          0 | PRIMARY      |            1 | PurchaseID   | A         |        1000 |     NULL |   NULL |      | BTREE      |         |               | YES     | NULL
|
| Purchases |          1 | NetID        |            1 | NetID        | A         |         411 |     NULL |   NULL | YES  | BTREE      |         |               | YES     | NULL       |
| Purchases |          1 | DishName     |            1 | DishName     | A         |         393 |     NULL |   NULL | YES  | BTREE      |         |               | YES     | NULL
|
| Purchases |          1 | RestaurantID |            1 | RestaurantID | A         |         323 |     NULL |   NULL |      | BTREE      |         |               | YES     | NULL
|
+-----------+------------+--------------+--------------+--------------+-----------+-------------+----------+--------+------+------------+---------+---------------+---------+------------+
4 rows in set (0.00 sec)

mysql> SHOW INDEX FROM Reviews;
+---------+------------+------------+--------------+-------------+-----------+-------------+----------+--------+------+------------+---------+---------------+---------+------------+
| Table   | Non_unique | Key_name   | Seq_in_index | Column_name | Collation | Cardinality | Sub_part | Packed | Null | Index_type | Comment |
Index_comment | Visible | Expression |
+---------+------------+------------+--------------+-------------+-----------+-------------+----------+--------+------+------------+---------+---------------+---------+------------+
| Reviews |          0 | PRIMARY    |            1 | ReviewID    | A         |        1000 |     NULL |   NULL |      | BTREE      |         |               | YES     | NULL       |
| Reviews |          1 | PurchaseID |            1 | PurchaseID  | A         |         624 |     NULL |   NULL |      | BTREE      |         |               | YES     | NULL       |
| Reviews |          1 | NetID      |            1 | NetID       | A         |         390 |     NULL |   NULL | YES  | BTREE      |         |               | YES     | NULL       |
+---------+------------+------------+--------------+-------------+-----------+-------------+----------+--------+------+------------+---------+---------------+---------+------------+
3 rows in set (0.00 sec)

mysql>
```

Default Query Performance

```
mysql> SHOW INDEX FROM  Dishes;+--------+------------+--------------+--------------+--------------+-----------+-------------+----------+--------+------+------------+---------+---------------+---------+------------+| Table  |
Non_unique | Key_name     | Seq_in_index | Column_name  | Collation | Cardinality | Sub_part | Packed | Null | Index_type | Comment | Index_comment | Visible | Expression
|+--------+------------+--------------+--------------+--------------+-----------+-------------+----------+--------+------+------------+---------+---------------+---------+------------+| Dishes |          0 | PRIMARY      |            1 | Name         | A
|        1000 |     NULL |   NULL |      | BTREE      |         |               | YES     | NULL  |
| Dishes |          1 | RestaurantID |            1 | RestaurantID | A         |          616 |     NULL |   NULL |      | BTREE      |         |               | YES     | NULL  || Dishes |          1 | Price_idx    |            1 | Price        | A         |
776 |     NULL |   NULL | YES  | BTREE      |         |               | YES     | NULL  |
+--------+------------+--------------+--------------+--------------+-----------+-------------+----------+--------+------+------------+---------+---------------+---------+------------+
3 rows in set (0.00 sec)

mysql> DROP INDEX Price_idx ON Dishes;
Query OK, 0 rows affected (0.02 sec)
Records: 0  Duplicates: 0  Warnings: 0
mysql> EXPLAIN ANALYZE SELECT r.name, AVG(d.Price) AS AvgPrice FROM Restaurant r JOIN Dishes d ON (r.restaurantID = d.RestaurantID) GROUP BY r.restaurantID;
+----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------
----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------+| EXPLAIN
|
+----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------
----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------+
| -> Table scan on <temporary>  (actual time=0.001..0.064 rows=616 loops=1)
    -> Aggregate using temporary table  (actual time=2.723..2.844 rows=616 loops=1)
        -> Nested loop inner join  (cost=451.25 rows=1000) (actual time=0.088..2.043 rows=1000 loops=1)
            -> Table scan on d  (cost=101.25 rows=1000) (actual time=0.047..0.368 rows=1000 loops=1)
            -> Single-row index lookup on r using PRIMARY (restaurantID=d.RestaurantID)  (cost=0.25 rows=1) (actual time=0.001..0.001 rows=1 loops=1000)
 |
+----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------
----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------+
1 row in set (0.00 sec)

mysql>
```

Index Design and Query Performance - 1

The following index design indexes on Price for Dishes relation. The execution time comparison from the default and current index design shows that the performance worsens after indexing on Price.

```
mysql> EXPLAIN ANALYZE SELECT r.name, AVG(d.Price) AS AvgPrice
    -> FROM Restaurant r JOIN Dishes d ON (r.restaurantID = d.RestaurantID)
    -> GROUP BY r.restaurantID;
+----------------------------------------------------------------------------------------------------------------------------------------------------------------
----------------------------------------------------------------------------------------------------------------------------------------------------------------
-----------------------------------------------------+
| EXPLAIN
|
+----------------------------------------------------------------------------------------------------------------------------------------------------------------
----------------------------------------------------------------------------------------------------------------------------------------------------------------
-----------------------------------------------------+
| -> Table scan on <temporary>  (actual time=0.002..0.066 rows=616 loops=1)
    -> Aggregate using temporary table  (actual time=8.936..9.059 rows=616 loops=1)
        -> Nested loop inner join  (cost=451.25 rows=1000) (actual time=0.564..7.724 rows=1000 loops=1)
            -> Table scan on d  (cost=101.25 rows=1000) (actual time=0.466..0.867 rows=1000 loops=1)
            -> Single-row index lookup on r using PRIMARY (restaurantID=d.RestaurantID)  (cost=0.25 rows=1) (actual time=0.006..0.006 rows=1 loops=1000)
 |
+----------------------------------------------------------------------------------------------------------------------------------------------------------------
----------------------------------------------------------------------------------------------------------------------------------------------------------------
-----------------------------------------------------+
1 row in set (0.01 sec)
```

Index Design and Query Performance - 2

The following index design indexes on restaurant name for Restaurant attribute. After comparing the execution time of this index design with the default, similar to the first index design, it can be concluded that the execution time worsens with an additional indexing column. The reason is that the additional indexing created unwanted complexity to the performance and since our query does not use any filtering or sorting attribute column that is not a primary or foreign key (which are already efficiently indexed by the database at the creation of relations) indexing on restaurant name is useless.

```
mysql> show index FROM Restaurant;
+------------+------------+----------+--------------+--------------+-----------+-------------+----------+--------+------+------------+---------+---------------+---------+------------+
| Table      | Non_unique | Key_name | Seq_in_index | Column_name  | Collation | Cardinality | Sub_part | Packed | Null | Index_type | Comment | Index_comment | Visible | Expression |
+------------+------------+----------+--------------+--------------+-----------+-------------+----------+--------+------+------------+---------+---------------+---------+------------+
| Restaurant |          0 | PRIMARY  |            1 | restaurantID | A         |        1136 |     NULL |   NULL |      | BTREE      |         |               | YES     | NULL       |
| Restaurant |          1 | name_idx |            1 | name         | A         |         967 |     NULL |   NULL | YES  | BTREE      |         |               | YES     | NULL       |
+------------+------------+----------+--------------+--------------+-----------+-------------+----------+--------+------+------------+---------+---------------+---------+------------+
2 rows in set (0.01 sec)

mysql> EXPLAIN ANALYZE SELECT r.name, AVG(d.Price) AS AvgPrice FROM Restaurant r JOIN Dishes d ON (r.restaurantID = d.RestaurantID) GROUP BY r.restaurantID;
+---------------------------------------------------------------------------------------------------------------------------------------------------------------------------+
| EXPLAIN
|
+---------------------------------------------------------------------------------------------------------------------------------------------------------------------------+
| -> Table scan on <temporary>  (actual time=0.001..0.072 rows=616 loops=1)
    -> Aggregate using temporary table  (actual time=3.499..3.629 rows=616 loops=1)
        -> Nested loop inner join  (cost=451.25 rows=1000) (actual time=0.050..2.660 rows=1000 loops=1)
            -> Table scan on d  (cost=101.25 rows=1000) (actual time=0.041..0.484 rows=1000 loops=1)
            -> Single-row index lookup on r using PRIMARY (restaurantID=d.RestaurantID)  (cost=0.25 rows=1) (actual time=0.002..0.002 rows=1 loops=1000)
 |
+---------------------------------------------------------------------------------------------------------------------------------------------------------------------------+
1 row in set (0.01 sec)

mysql>
```

Index Design and Query Performance - 3

Similar to first and second index designs, this index design also contributes to no improvement in performance. One potential factor that contributes to longer execution times for this and the previous two index designs is the planning time that the database optimizer takes to efficiently perform query execution. WIth additional indexing, the storage for additional indexing and the time for analyzing those potential execution paths increases.

```
mysql> SHOW INDEX FROM Dishes;
+--------+------------+----------------------------+--------------+--------------+-----------+-------------+----------+--------+------+------------+---------+---------------+---------+------------+
| Table  | Non_unique | Key_name                   | Seq_in_index | Column_name  | Collation | Cardinality | Sub_part | Packed | Null | Index_type | Comment | Index_comment | Visible | Expression |
+--------+------------+----------------------------+--------------+--------------+-----------+-------------+----------+--------+------+------------+---------+---------------+---------+------------+
| Dishes |          0 | PRIMARY                    |            1 | Name         | A         |        1000 |     NULL |   NULL |      | BTREE      |         |               | YES     | NULL       |
| Dishes |          1 | RestaurantID               |            1 | RestaurantID | A         |         616 |     NULL |   NULL |      | BTREE      |         |               | YES     | NULL       |
| Dishes |          1 | restaurantID_and_Price_idx |            1 | RestaurantID | A         |         616 |     NULL |   NULL |      | BTREE      |         |               | YES     | NULL       |
| Dishes |          1 | restaurantID_and_Price_idx |            2 | Price        | A         |        1000 |     NULL |   NULL | YES  | BTREE      |         |               | YES     | NULL       |
+--------+------------+----------------------------+--------------+--------------+-----------+-------------+----------+--------+------+------------+---------+---------------+---------+------------+
4 rows in set (0.00 sec)

mysql>
mysql>
mysql> EXPLAIN ANALYZE SELECT r.name, AVG(d.Price) AS AvgPrice FROM Restaurant r JOIN Dishes d ON (r.restaurantID = d.RestaurantID) GROUP BY r.restaurantID;
+---------------------------------------------------------------------------------------------------------------------------------------------------------------------------+
| EXPLAIN
|
+---------------------------------------------------------------------------------------------------------------------------------------------------------------------------+
| -> Table scan on <temporary>  (actual time=0.002..0.058 rows=616 loops=1)
    -> Aggregate using temporary table  (actual time=2.229..2.343 rows=616 loops=1)
        -> Nested loop inner join  (cost=451.25 rows=1000) (actual time=0.093..2.060 rows=1000 loops=1)
            -> Index scan on d using restaurantID_and_Price_idx  (cost=101.25 rows=1000) (actual time=0.081..0.458 rows=1000 loops=1)
            -> Single-row index lookup on r using PRIMARY (restaurantID=d.RestaurantID)  (cost=0.25 rows=1) (actual time=0.001..0.001 rows=1 loops=1000)
 |
+---------------------------------------------------------------------------------------------------------------------------------------------------------------------------+
1 row in set (0.01 sec)

mysql>
```

Second Advanced Query

SELECT Temp.name, d.Name
*FROM (SELECT * FROM Restaurant r WHERE r.cuisine = “Italian”) AS Temp*
JOIN Dishes d ON (Temp.restaurantID = d.RestaurantID) JOIN Purchases p ON
(d.RestaurantID = p.RestaurantID) JOIN Reviews ON (p.NetID = Reviews.NetID)
WHERE Reviews.Rating >= 4.0;

<u>Default Index Designs</u>

```
mysql> SHOW INDEX FROM Restaurant;
+------------+------------+----------+--------------+--------------+-----------+-------------+----------+--------+------+------------+---------+---------------+---------+------------+
| Table      | Non_unique | Key_name | Seq_in_index | Column_name  | Collation | Cardinality | Sub_part | Packed | Null | Index_type | Comment |
Index_comment | Visible | Expression |
+------------+------------+----------+--------------+--------------+-----------+-------------+----------+--------+------+------------+---------+---------------+---------+------------+
| Restaurant |          0 | PRIMARY  |            1 | restaurantID | A         |        1136 |     NULL |   NULL |      | BTREE      |         |               | YES     | NULL       |
+------------+------------+----------+--------------+--------------+-----------+-------------+----------+--------+------+------------+---------+---------------+---------+------------+
1 row in set (0.01 sec)

mysql> SHOW INDEX FROM Dishes;
+--------+------------+--------------+--------------+--------------+-----------+-------------+----------+--------+------+------------+---------+---------------+---------+------------+
| Table  | Non_unique | Key_name     | Seq_in_index | Column_name  | Collation | Cardinality | Sub_part | Packed | Null | Index_type | Comment |
Index_comment | Visible | Expression |
+--------+------------+--------------+--------------+--------------+-----------+-------------+----------+--------+------+------------+---------+---------------+---------+------------+
| Dishes |          0 | PRIMARY      |            1 | Name         | A         |        1000 |     NULL |   NULL |      | BTREE      |         |               | YES     | NULL       |
| Dishes |          1 | RestaurantID |            1 | RestaurantID | A         |         616 |     NULL |   NULL |      | BTREE      |         |               | YES     | NULL       |
+--------+------------+--------------+--------------+--------------+-----------+-------------+----------+--------+------+------------+---------+---------------+---------+------------+
2 rows in set (0.01 sec)

mysql> SHOW INDEX FROM Purchases;
+-----------+------------+--------------+--------------+--------------+-----------+-------------+----------+--------+------+------------+---------+---------------+---------+------------+
| Table     | Non_unique | Key_name     | Seq_in_index | Column_name  | Collation | Cardinality | Sub_part | Packed | Null | Index_type | Comment |
Index_comment | Visible | Expression |
+-----------+------------+--------------+--------------+--------------+-----------+-------------+----------+--------+------+------------+---------+---------------+---------+------------+
| Purchases |          0 | PRIMARY      |            1 | PurchaseID   | A         |        1000 |     NULL |   NULL |      | BTREE      |         |               | YES     | NULL
|
| Purchases |          1 | NetID        |            1 | NetID        | A         |         411 |     NULL |   NULL | YES  | BTREE      |         |               | YES     | NULL       |
| Purchases |          1 | DishName     |            1 | DishName     | A         |         393 |     NULL |   NULL | YES  | BTREE      |         |               | YES     | NULL
|
| Purchases |          1 | RestaurantID |            1 | RestaurantID | A         |         323 |     NULL |   NULL |      | BTREE      |         |               | YES     | NULL
|
+-----------+------------+--------------+--------------+--------------+-----------+-------------+----------+--------+------+------------+---------+---------------+---------+------------+
4 rows in set (0.00 sec)

mysql> SHOW INDEX FROM Reviews;
+---------+------------+------------+--------------+-------------+-----------+-------------+----------+--------+------+------------+---------+---------------+---------+------------+
| Table   | Non_unique | Key_name   | Seq_in_index | Column_name | Collation | Cardinality | Sub_part | Packed | Null | Index_type | Comment |
Index_comment | Visible | Expression |
+---------+------------+------------+--------------+-------------+-----------+-------------+----------+--------+------+------------+---------+---------------+---------+------------+
| Reviews |          0 | PRIMARY    |            1 | ReviewID    | A         |        1000 |     NULL |   NULL |      | BTREE      |         |               | YES     | NULL       |
| Reviews |          1 | PurchaseID |            1 | PurchaseID  | A         |         624 |     NULL |   NULL |      | BTREE      |         |               | YES     | NULL       |
| Reviews |          1 | NetID      |            1 | NetID       | A         |         390 |     NULL |   NULL | YES  | BTREE      |         |               | YES     | NULL       |
+---------+------------+------------+--------------+-------------+-----------+-------------+----------+--------+------+------------+---------+---------------+---------+------------+
3 rows in set (0.00 sec)
mysql>
```

Default Query Performance

```
mysql> EXPLAIN ANALYZE SELECT Temp.name, d.Name  FROM (SELECT * FROM Restaurant r WHERE r.cuisine = "Italian") AS Temp JOIN Dishes
d ON (Temp.restaurantID = d.RestaurantID) JOIN Purchases p ON (d.RestaurantID = p.RestaurantID) JOIN Reviews ON (p.NetID = Reviews.NetID)
WHERE Reviews.Rating >= 4.0;
+------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------
-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------
-------------+
| EXPLAIN
|
+------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------
-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------
-------------+
| -> Nested loop inner join  (cost=611.31 rows=139) (actual time=0.551..5.080 rows=63 loops=1)
    -> Nested loop inner join  (cost=540.99 rows=85) (actual time=0.503..4.843 rows=26 loops=1)
        -> Nested loop inner join  (cost=451.25 rows=100) (actual time=0.215..2.302 rows=75 loops=1)
            -> Filter: (p.NetID is not null)  (cost=101.25 rows=1000) (actual time=0.105..0.587 rows=1000 loops=1)
                -> Table scan on p  (cost=101.25 rows=1000) (actual time=0.104..0.433 rows=1000 loops=1)
            -> Filter: (r.cuisine = 'Italian')  (cost=0.25 rows=0) (actual time=0.002..0.002 rows=0 loops=1000)
                -> Single-row index lookup on r using PRIMARY (restaurantID=p.RestaurantID)  (cost=0.25 rows=1) (actual time=0.001..0.001 rows=1
loops=1000)
        -> Filter: (Reviews.Rating >= 4)  (cost=0.64 rows=1) (actual time=0.033..0.034 rows=0 loops=75)
            -> Index lookup on Reviews using NetID (NetID=p.NetID)  (cost=0.64 rows=3) (actual time=0.031..0.033 rows=2 loops=75)
    -> Index lookup on d using RestaurantID (RestaurantID=p.RestaurantID)  (cost=0.66 rows=2) (actual time=0.008..0.009 rows=2 loops=26)
 |
+------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------
-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------
-------------+
1 row in set (0.01 sec)
mysql>
```

Index Design and Query Performance -1

The following explain analysis indexes on type of cuisine for Restaurant relation. The reason for indexing on cuisine is because the advanced query filters an arbitrary cuisine by performing a table scan which is linear in time complexity. In expectation, indexing on cuisine should result in shorter execution time as the scan performance is logarithmic. However, after comparing with the default indexing scheme, we notice that there is no improvement depicted, which potentially means that the database automatically performs non-clustered indexing based on its statistics.

```
mysql> SHOW INDEX FROM Restaurant;
+------------+------------+-------------+--------------+-------------+-----------+-------------+----------+--------+------+------------+---------+---------------+---------+------------+
| Table      | Non_unique | Key_name    | Seq_in_index | Column_name  | Collation | Cardinality | Sub_part | Packed | Null | Index_type | Comment | Index_comment | Visible |
Expression |
+------------+------------+-------------+--------------+-------------+-----------+-------------+----------+--------+------+------------+---------+---------------+---------+------------+
| Restaurant |          0 | PRIMARY     |            1 | restaurantID | A         |        1136 |     NULL |   NULL |      | BTREE      |         |               | YES     | NULL       |
| Restaurant |          1 | cuisine_idx |            1 | cuisine      | A         |          11 |     NULL |   NULL | YES  | BTREE      |         |               | YES     | NULL       |
+------------+------------+-------------+--------------+-------------+-----------+-------------+----------+--------+------+------------+---------+---------------+---------+------------+
2 rows in set (0.01 sec)

mysql> EXPLAIN ANALYZE SELECT Temp.name, d.Name  FROM (SELECT * FROM Restaurant r WHERE r.cuisine = "Italian") AS Temp JOIN Dishes d ON
(Temp.restaurantID = d.RestaurantID) JOIN Purchases p ON (d.RestaurantID = p.RestaurantID) JOIN Reviews ON (p.NetID = Reviews.NetID) WHERE Reviews.Rating >= 4.0;
+------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------
-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------+
| EXPLAIN
|
+------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------
-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------+
| -> Nested loop inner join  (cost=587.92 rows=118) (actual time=0.353..4.795 rows=63 loops=1)
    -> Nested loop inner join  (cost=527.88 rows=73) (actual time=0.344..4.720 rows=26 loops=1)
        -> Nested loop inner join  (cost=451.25 rows=85) (actual time=0.314..4.226 rows=75 loops=1)
            -> Filter: (p.NetID is not null)  (cost=101.25 rows=1000) (actual time=0.151..0.566 rows=1000 loops=1)
                -> Table scan on p  (cost=101.25 rows=1000) (actual time=0.150..0.423 rows=1000 loops=1)
        -> Filter: (r.cuisine = 'Italian')  (cost=0.25 rows=0) (actual time=0.003..0.003 rows=0 loops=1000)
            -> Single-row index lookup on r using PRIMARY (restaurantID=p.RestaurantID)  (cost=0.25 rows=1) (actual time=0.003..0.003 rows=1 loops=1000)
    -> Filter: (Reviews.Rating >= 4)  (cost=0.64 rows=1) (actual time=0.006..0.006 rows=0 loops=75)
        -> Index lookup on Reviews using NetID (NetID=p.NetID)  (cost=0.64 rows=3) (actual time=0.005..0.006 rows=2 loops=75)
  -> Index lookup on d using RestaurantID (RestaurantID=p.RestaurantID)  (cost=0.66 rows=2) (actual time=0.002..0.002 rows=2 loops=26)
 |
+---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------
-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------+
1 row in set (0.01 sec)
```

Mysql

Index Design and Query Performance - 2

The second index design indexes on rating for Reviews relations. After comparing the execution times for current and default index designs, we found that the current index design overall gives slower execution time.

```
mysql> SHOW INDEX FROM Reviews;
+---------+------------+------------+--------------+-------------+-----------+-------------+----------+--------+------+------------+---------+---------------+---------+------------+
| Table   | Non_unique | Key_name   | Seq_in_index | Column_name | Collation | Cardinality | Sub_part | Packed | Null | Index_type | Comment | Index_comment | Visible |
Expression |
+---------+------------+------------+--------------+-------------+-----------+-------------+----------+--------+------+------------+---------+---------------+---------+------------+
| Reviews |          0 | PRIMARY    |            1 | ReviewID    | A         |        1000 |     NULL |   NULL |      | BTREE      |         |               | YES     | NULL       |
| Reviews |          1 | PurchaseID |            1 | PurchaseID  | A         |         624 |     NULL |   NULL |      | BTREE      |         |               | YES     | NULL       |
| Reviews |          1 | NetID      |            1 | NetID       | A         |         390 |     NULL |   NULL | YES  | BTREE      |         |               | YES     | NULL       |
| Reviews |          1 | rating_idx |            1 | Rating      | A         |          51 |     NULL |   NULL | YES  | BTREE      |         |               | YES     | NULL       |
+---------+------------+------------+--------------+-------------+-----------+-------------+----------+--------+------+------------+---------+---------------+---------+------------+
4 rows in set (0.00 sec)

mysql> EXPLAIN ANALYZE SELECT Temp.name, d.Name  FROM (SELECT * FROM Restaurant r WHERE r.cuisine = "Italian") AS Temp JOIN Dishes d ON
(Temp.restaurantID = d.RestaurantID) JOIN Purchases p ON (d.RestaurantID = p.RestaurantID) JOIN Reviews ON (p.NetID = Reviews.NetID) WHERE Reviews.Rating >= 4.0;
+---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------
-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------+
| EXPLAIN
|
+---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------
-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------+
| -> Nested loop inner join  (cost=452.10 rows=76) (actual time=1.746..8.117 rows=63 loops=1)
    -> Nested loop inner join  (cost=413.67 rows=47) (actual time=1.736..8.028 rows=26 loops=1)
        -> Nested loop inner join  (cost=250.16 rows=467) (actual time=1.349..7.058 rows=454 loops=1)
            -> Filter: (Reviews.NetID is not null)  (cost=86.66 rows=192) (actual time=1.204..2.098 rows=192 loops=1)
                -> Index range scan on Reviews using rating_idx, with index condition: (Reviews.Rating >= 4)  (cost=86.66 rows=192) (actual time=1.134..1.995 rows=192 loops=1)
            -> Index lookup on p using NetID (NetID=Reviews.NetID)  (cost=0.61 rows=2) (actual time=0.023..0.025 rows=2 loops=192)
        -> Filter: (r.cuisine = 'Italian')  (cost=0.25 rows=0) (actual time=0.002..0.002 rows=0 loops=454)
            -> Single-row index lookup on r using PRIMARY (restaurantID=p.RestaurantID)  (cost=0.25 rows=1) (actual time=0.001..0.002 rows=1 loops=454)
    -> Index lookup on d using RestaurantID (RestaurantID=p.RestaurantID)  (cost=0.66 rows=2) (actual time=0.002..0.003 rows=2 loops=26)
 |
+---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------
------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------
----------------------------------+
1 row in set (0.01 sec)
mysql>
```

Index Design and Query Performance - 3

The third index design indexes on two attributes for two different relations, which combines the last two index designs: indexing on cuisine and ratings for Restaurant and Reviews respectively. We can observe that the combination of index definitions comprehensively outputs better execution time for JOINS. Hence this index design can be an acceptable index design for improving query performance.

```
mysql> SHOW INDEX FROM Restaurant;
+------------+------------+-------------+--------------+--------------+-----------+-------------+----------+--------+------+------------+---------+---------------+---------+------------+
| Table      | Non_unique | Key_name    | Seq_in_index | Column_name  | Collation | Cardinality | Sub_part | Packed | Null | Index_type | Comment | Index_comment | Visible |
Expression |
+------------+------------+-------------+--------------+--------------+-----------+-------------+----------+--------+------+------------+---------+---------------+---------+------------+
| Restaurant |          0 | PRIMARY     |            1 | restaurantID | A         |        1136 |     NULL |   NULL |      | BTREE      |         |               | YES     | NULL       |
| Restaurant |          1 | cuisine_idx |            1 | cuisine      | A         |          11 |     NULL |   NULL | YES  | BTREE      |         |               | YES     | NULL       |
+------------+------------+-------------+--------------+--------------+-----------+-------------+----------+--------+------+------------+---------+---------------+---------+------------+
2 rows in set (0.00 sec)

mysql> SHOW INDEX FROM Reviews;
+---------+------------+------------+--------------+-------------+-----------+-------------+----------+--------+------+------------+---------+---------------+---------+------------+
| Table   | Non_unique | Key_name   | Seq_in_index | Column_name | Collation | Cardinality | Sub_part | Packed | Null | Index_type | Comment | Index_comment | Visible |
Expression|
+---------+------------+------------+--------------+-------------+-----------+-------------+----------+--------+------+------------+---------+---------------+---------+------------+
| Reviews |          0 | PRIMARY    |            1 | ReviewID    | A         |        1000 |     NULL |   NULL |      | BTREE      |         |               | YES     | NULL|
| Reviews |          1 | PurchaseID |            1 | PurchaseID  | A         |         624 |     NULL |   NULL |      | BTREE      |         |               | YES     | NULL|
| Reviews |          1 | NetID      |            1 | NetID       | A         |         390 |     NULL |   NULL | YES  | BTREE      |         |               | YES     | NULL|
| Reviews |          1 | rating_idx |            1 | Rating      | A         |          51 |     NULL |   NULL | YES  | BTREE      |         |               | YES     | NULL|
+---------+------------+------------+--------------+-------------+-----------+-------------+----------+--------+------+------------+---------+---------------+---------+------------+
4 rows in set (0.00 sec)

mysql> EXPLAIN ANALYZE SELECT Temp.name, d.Name  FROM (SELECT * FROM Restaurant r WHERE r.cuisine = "Italian") AS Temp JOIN Dishes d ON
(Temp.restaurantID = d.RestaurantID) JOIN Purchases p ON (d.RestaurantID = p.RestaurantID) JOIN Reviews ON (p.NetID = Reviews.NetID) WHERE Reviews.Rating >= 4.0;
+---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------
---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------+
| EXPLAIN
|
+---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------
----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------
----------------------------------------------------------------------------------------------------------------------------------------------------+
| -> Nested loop inner join  (cost=446.49 rows=65) (actual time=0.260..2.358 rows=63 loops=1)
    -> Nested loop inner join  (cost=413.67 rows=40) (actual time=0.252..2.284 rows=26 loops=1)
        -> Nested loop inner join  (cost=250.16 rows=467) (actual time=0.172..1.464 rows=454 loops=1)
            -> Filter: (Reviews.NetID is not null)  (cost=86.66 rows=192) (actual time=0.158..0.337 rows=192 loops=1)
                -> Index range scan on Reviews using rating_idx, with index condition: (Reviews.Rating >= 4)  (cost=86.66 rows=192) (actual time=0.157..0.307 rows=192 loops=1)
            -> Index lookup on p using NetID (NetID=Reviews.NetID)  (cost=0.61 rows=2) (actual time=0.005..0.005 rows=2 loops=192)
        -> Filter: (r.cuisine = 'Italian')  (cost=0.25 rows=0) (actual time=0.002..0.002 rows=0 loops=454)
            -> Single-row index lookup on r using PRIMARY (restaurantID=p.RestaurantID)  (cost=0.25 rows=1) (actual time=0.001..0.001 rows=1 loops=454)
    -> Index lookup on d using RestaurantID (RestaurantID=p.RestaurantID)  (cost=0.66 rows=2) (actual time=0.002..0.002 rows=2 loops=26)
 |
+---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------
----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------
-----------------------------------------------------------------------------------------------------------------------------------------------------+
1 row in set (0.01 sec)
mysql>
```