Krishna Konda, Praveen Kalva, Justin Ansell, Wyatt Sass

CS 411 Database Systems

TeamName: PreQL

Stage 3 Database Implementation and Indexing - PreQL

Data Definition Language (DDL) commands

We used the following SQL commands to create our tables:

```
CREATE TABLE UserProfile(
     Username VARCHAR(255),
     Password VARCHAR(255),
     Age INT,
     OwnsMac BOOLEAN,
     OwnsLinux BOOLEAN,
     OwnsWindows BOOLEAN,
     FavoriteGenres VARCHAR(1000),
     PRIMARY KEY(Username)
);
CREATE TABLE GameInfo(
     GameID INT,
     GameName VARCHAR(255),
     Attributes VARCHAR(1000),
     Description VARCHAR(1000),
     ReleaseDate VARCHAR(15),
     PlatformMac BOOLEAN,
     PlatformWindows BOOLEAN,
     PlatformLinux BOOLEAN,
```

```
    Price INT,
    RequiredAge INT,
    MetaCritic INT,
    PlayerEstimate INT,
    BackgroundImageURL VARCHAR(255),
    HeaderImageURL VARCHAR(255),
    PRIMARY KEY(GameID)
);
CREATE TABLE PlayTime(
    Username VARCHAR(255),
    GameID INT,
    HoursPlayed INT,
    PRIMARY KEY(Username, GameID),
    FOREIGN KEY(Username) REFERENCES UserProfile(Username),
    FOREIGN KEY(GameID) REFERENCES GameInfo(GameID)
);
CREATE TABLE Comments(
    CommentID INT,
    GameID INT,
    Username VARCHAR(255),
    CommentText VARCHAR(2000),
    PRIMARY KEY(CommentID),
    FOREIGN KEY(Username) REFERENCES UserProfile(Username),
    FOREIGN KEY(GameID) REFERENCES GameInfo(GameID)
);
CREATE TABLE Rating(
```

```
    Vote INT,
    Username VARCHAR(255),
    GameID INT,
    PRIMARY KEY(Username, GameID),
    FOREIGN KEY(Username) REFERENCES UserProfile(Username),
    FOREIGN KEY(GameID) REFERENCES GameInfo(GameID)
);
```

Tables:

The tables we created are shown in the screenshot below. We implemented 5 different tables: UserProfile, GameInfo, Rating, PlayTime, and Comments.

```
mysql> show tables;
+-----------------+
| Tables_in_preql |
+-----------------+
| Comments        |
| GameInfo        |
| PlayTime        |
| Rating          |
| UserProfile     |
+-----------------+
5 rows in set (0.00 sec)
```

The following screenshots show the number of rows of data we have for each table. The GameInfo table was created using filtered data from a real dataset, while the other tables were user-generated data.

```
mysql> select count(*) from GameInfo;
+----------+
| count(*) |
+----------+
|    13304 |
+----------+
1 row in set (0.03 sec)
```

```
mysql> select count(*) from UserProfile;
+----------+
| count(*) |
+----------+
|     1001 |
+----------+
1 row in set (0.01 sec)
```

```
mysql> select count(*) from Rating;
+----------+
| count(*) |
+----------+
|     1001 |
+----------+
1 row in set (0.03 sec)
```

```
mysql> select count(*) from PlayTime;
+----------+
| count(*) |
+----------+
|     1001 |
+----------+
1 row in set (0.00 sec)
```

```
mysql> select count(*) from Comments;
+----------+
| count(*) |
+----------+
|     1000 |
+----------+
1 row in set (0.01 sec)
```

Advanced Queries:

1.Popular Games by Young Adults:

SELECT GameID, GameName, SUM(HoursPlayed) as totalHours

FROM PlayTime NATURAL JOIN GameInfo NATURAL JOIN UserProfile

WHERE Age > 16 AND AGE < 20

GROUP BY GameID, GameName

ORDER BY totalHours DESC, GameName ASC;

Top 15 Results for query 1:

```
mysql> SELECT GameID, GameName, SUM(HoursPlayed) as totalHours
    -> FROM PlayTime NATURAL JOIN GameInfo NATURAL JOIN UserProfile
    -> WHERE Age > 16 AND AGE < 20
    -> GROUP BY GameID, GameName
    -> ORDER BY totalHours DESC, GameName ASC
    -> LIMIT 15;
+--------+-----------------------------------------------+------------+
| GameID | GameName                                      | totalHours |
+--------+-----------------------------------------------+------------+
| 403640 | Dishonored 2                                  |        985 |
| 341680 | stratO                                        |        984 |
| 389850 | Legacy of Dorn: Herald of Oblivion            |        977 |
| 429200 | Super Helmets on Fire DX Ultra Edition Plus Alpha |        976 |
| 262260 | Jets'n'Guns Gold                              |        909 |
| 384570 | ZanZarah: The Hidden Portal                   |        902 |
|  46520 | Wasteland Angel                               |        890 |
| 364790 | Voyage to Farland                             |        887 |
| 367110 | Dustbowl                                      |        856 |
| 520766 | Naruto Shippuden Uncut: The Unfinished Page    |        848 |
| 402150 | String Theory                                 |        847 |
|  29180 | Osmos                                         |        818 |
| 523170 | ShotForge                                     |        811 |
| 533700 | VR Retreat                                    |        805 |
| 214360 | Tower Wars                                    |        792 |
+--------+-----------------------------------------------+------------+
15 rows in set (0.00 sec)
```

Indexing Optimization for query 1:

Default Indices:

```
----------------------------------+
| -> Sort: totalHours DESC, GameInfo.GameName  (actual time=1.531..1.540 rows=66 loops=1)
    -> Table scan on <temporary>  (actual time=1.398..1.411 rows=66 loops=1)
        -> Aggregate using temporary table  (actual time=1.395..1.395 rows=66 loops=1)
            -> Nested loop inner join  (cost=180.44 rows=111) (actual time=0.122..1.298 rows=66 loops=1)
                -> Nested loop inner join  (cost=141.52 rows=111) (actual time=0.110..0.899 rows=66 loops=1)
                    -> Filter: ((UserProfile.Age > 16) and (UserProfile.Age < 20))  (cost=102.60 rows=111) (actual time=0.073..0.587 rows=66 loops
=1)
                        -> Table scan on UserProfile  (cost=102.60 rows=1001) (actual time=0.065..0.449 rows=1001 loops=1)
                    -> Index lookup on PlayTime using PRIMARY (Username=UserProfile.Username)  (cost=0.25 rows=1) (actual time=0.004..0.004 rows=1
 loops=66)
                -> Single-row index lookup on GameInfo using PRIMARY (GameID=PlayTime.GameID)  (cost=0.25 rows=1) (actual time=0.006..0.006 rows=1
 loops=66)
 |
+---------------------------------------------------------------------------------------------------------------------------------------------------
---------------------------------------------------------------------------------------------------------------------------------------------------
---------------------------------------------------------------------------------------------------------------------------------------------------
---------------------------------------------------------------------------------------------------------------------------------------------------
---------------------------------------------------------------------------------------------------------------------------------------------------
---------------------------------------------------------------------------------------------------------------------------------------------------
---------------------------------------------------------------------------------------------------------------------------------------------------
----------------------------------+
1 row in set (0.01 sec)
```

Ran 0.01 sec.

With Age index:

```
mysql> CREATE INDEX idx_Age ON UserProfile(Age);
Query OK, 0 rows affected (0.07 sec)
Records: 0  Duplicates: 0  Warnings: 0

mysql> SHOW INDEX FROM UserProfile;
+-------------+------------+----------+--------------+-------------+-----------+-------------+----------+--------+------+------------+---------+--
-------------+---------+------------+
| Table       | Non_unique | Key_name | Seq_in_index | Column_name | Collation | Cardinality | Sub_part | Packed | Null | Index_type | Comment | I
ndex_comment | Visible | Expression |
+-------------+------------+----------+--------------+-------------+-----------+-------------+----------+--------+------+------------+---------+--
-------------+---------+------------+
| UserProfile |          0 | PRIMARY  |            1 | Username    | A         |        1001 |     NULL |   NULL |      | BTREE      |         |
             | YES     | NULL       |
| UserProfile |          1 | idx_Age  |            1 | Age         | A         |          41 |     NULL |   NULL | YES  | BTREE      |         |
             | YES     | NULL       |
+-------------+------------+----------+--------------+-------------+-----------+-------------+----------+--------+------+------------+---------+--
-------------+---------+------------+
2 rows in set (0.01 sec)
```

```
-------------------------------------------------------------------------+
| -> Sort: totalHours DESC, GameInfo.GameName  (actual time=0.946..0.955 rows=66 loops=1)
    -> Table scan on <temporary>  (actual time=0.714..0.728 rows=66 loops=1)
        -> Aggregate using temporary table  (actual time=0.712..0.712 rows=66 loops=1)
            -> Nested loop inner join  (cost=63.73 rows=66) (actual time=0.040..0.627 rows=66 loops=1)
                -> Nested loop inner join  (cost=40.63 rows=66) (actual time=0.029..0.330 rows=66 loops=1)
                    -> Filter: ((UserProfile.Age > 16) and (UserProfile.Age < 20))  (cost=17.53 rows=66) (actual time=0.016..0.068 rows=66 loops=1
)
                        -> Covering index range scan on UserProfile using idx_Age over (16 < Age < 20)  (cost=17.53 rows=66) (actual time=0.013..0
.051 rows=66 loops=1)
                    -> Index lookup on PlayTime using PRIMARY (Username=UserProfile.Username)  (cost=0.25 rows=1) (actual time=0.003..0.004 rows=1
 loops=66)
                -> Single-row index lookup on GameInfo using PRIMARY (GameID=PlayTime.GameID)  (cost=0.25 rows=1) (actual time=0.004..0.004 rows=1
 loops=66)
 |
+-------------------------------------------------------------------------+
1 row in set (0.01 sec)
```

Ran in 0.01 sec. The costs all around are much better than the default indices.

With HoursPlayed index:

```
mysql> CREATE INDEX idx_HoursPlayed ON PlayTime(HoursPlayed);
Query OK, 0 rows affected (0.08 sec)
Records: 0  Duplicates: 0  Warnings: 0

mysql> SHOW INDEX FROM PlayTime;
+----------+------------+-----------------+--------------+-------------+-----------+-------------+----------+--------+------+------------+--------
-+---------------+---------+------------+
| Table    | Non_unique | Key_name        | Seq_in_index | Column_name | Collation | Cardinality | Sub_part | Packed | Null | Index_type | Comment
 | Index_comment | Visible | Expression |
+----------+------------+-----------------+--------------+-------------+-----------+-------------+----------+--------+------+------------+--------
-+---------------+---------+------------+
| PlayTime |          0 | PRIMARY         |            1 | Username    | A         |        1001 |     NULL |   NULL |      | BTREE      |
 |               | YES     | NULL       |
| PlayTime |          0 | PRIMARY         |            2 | GameID      | A         |        1001 |     NULL |   NULL |      | BTREE      |
 |               | YES     | NULL       |
| PlayTime |          1 | GameID          |            1 | GameID      | A         |         972 |     NULL |   NULL |      | BTREE      |
 |               | YES     | NULL       |
| PlayTime |          1 | idx_HoursPlayed |            1 | HoursPlayed | A         |         634 |     NULL |   NULL | YES  | BTREE      |
 |               | YES     | NULL       |
+----------+------------+-----------------+--------------+-------------+-----------+-------------+----------+--------+------+------------+--------
-+---------------+---------+------------+
4 rows in set (0.01 sec)
```

```
---------------------------------+
| -> Sort: totalHours DESC, GameInfo.GameName  (actual time=1.362..1.370 rows=66 loops=1)
    -> Table scan on <temporary>  (actual time=1.290..1.304 rows=66 loops=1)
        -> Aggregate using temporary table  (actual time=1.289..1.289 rows=66 loops=1)
            -> Nested loop inner join  (cost=180.44 rows=111) (actual time=0.143..1.200 rows=66 loops=1)
                -> Nested loop inner join  (cost=141.52 rows=111) (actual time=0.132..0.893 rows=66 loops=1)
                    -> Filter: ((UserProfile.Age > 16) and (UserProfile.Age < 20))  (cost=102.60 rows=111) (actual time=0.110..0.590 rows=66 loops
=1)
                        -> Table scan on UserProfile  (cost=102.60 rows=1001) (actual time=0.102..0.460 rows=1001 loops=1)
                    -> Index lookup on PlayTime using PRIMARY (Username=UserProfile.Username)  (cost=0.25 rows=1) (actual time=0.003..0.004 rows=1
 loops=66)
                -> Single-row index lookup on GameInfo using PRIMARY (GameID=PlayTime.GameID)  (cost=0.25 rows=1) (actual time=0.004..0.004 rows=1
 loops=66)
 |
+---------------------------------+
1 row in set (0.01 sec)
```

Ran in 0.01 sec. No improvement in costs.

With GameName index:

```
mysql> CREATE INDEX idx_GameName ON GameInfo(GameName);
Query OK, 0 rows affected (0.22 sec)
Records: 0  Duplicates: 0  Warnings: 0

mysql> SHOW INDEX FROM GameInfo;
+----------+------------+--------------------+--------------+----------------+-----------+-------------+----------+--------+------+------------+---------
------+---------------+---------+------------+
| Table    | Non_unique | Key_name           | Seq_in_index | Column_name    | Collation | Cardinality | Sub_part | Packed | Null | Index_type | C
omment | Index_comment | Visible | Expression |
+----------+------------+--------------------+--------------+----------------+-----------+-------------+----------+--------+------+------------+---------
------+---------------+---------+------------+
| GameInfo |          0 | PRIMARY            |            1 | GameID         | A         |       11106 |     NULL |   NULL |      | BTREE      |
       |               | YES     | NULL       |
| GameInfo |          1 | idx_PlayerEstimate |            1 | PlayerEstimate | A         |        1915 |     NULL |   NULL | YES  | BTREE      |
       |               | YES     | NULL       |
| GameInfo |          1 | idx_GameName       |            1 | GameName       | A         |       11106 |     NULL |   NULL | YES  | BTREE      |
       |               | YES     | NULL       |
+----------+------------+--------------------+--------------+----------------+-----------+-------------+----------+--------+------+------------+---------
------+---------------+---------+------------+
3 rows in set (0.00 sec)
```

```
---------------------------------+
| -> Sort: totalHours DESC, GameInfo.GameName  (actual time=1.454..1.463 rows=66 loops=1)
    -> Table scan on <temporary>  (actual time=1.372..1.386 rows=66 loops=1)
        -> Aggregate using temporary table  (actual time=1.370..1.370 rows=66 loops=1)
            -> Nested loop inner join  (cost=180.44 rows=111) (actual time=0.108..1.234 rows=66 loops=1)
                -> Nested loop inner join  (cost=141.52 rows=111) (actual time=0.096..0.921 rows=66 loops=1)
                    -> Filter: ((UserProfile.Age > 16) and (UserProfile.Age < 20))  (cost=102.60 rows=111) (actual time=0.079..0.616 rows=66 loops
=1)
                        -> Table scan on UserProfile  (cost=102.60 rows=1001) (actual time=0.069..0.462 rows=1001 loops=1)
                    -> Index lookup on PlayTime using PRIMARY (Username=UserProfile.Username)  (cost=0.25 rows=1) (actual time=0.003..0.004 rows=1
 loops=66)
                -> Single-row index lookup on GameInfo using PRIMARY (GameID=PlayTime.GameID)  (cost=0.25 rows=1) (actual time=0.004..0.004 rows=1
 loops=66)
 |
+---------------------------------+
---------------------------------------------------------------------------------------------------------------------------------------------------------
---------------------------------------------------------------------------------------------------------------------------------------------------------
---------------------------------------------------------------------------------------------------------------------------------------------------------
---------------------------------------------------------------------------------------------------------------------------------------------------------
---------------------------------------------------------------------------------------------------------------------------------------------------------
---------------------------------------------------------------------------------------------------------------------------------------------------------
---------------------------------------------------------------------------------------------------------------------------------------------------------
---------------------------------+
1 row in set (0.00 sec)
```

Ran in 0.00 sec. Costs are generally the same as default.

We think adding an Age index improved the costs because our query involves a condition on the age to check if the user is within an age range. Comparing costs, we can see that the age index outperformed the default indices and the other indices(HoursPlayed and GameName) did not improve costs very much. Because of this, we decided to keep the Age index so our query can run with lower cost.

2.Highest Rated and Highest Player Estimate Games:

SELECT g.GameID, g.GameName, g.PlayerEstimate, g.Metacritic, g.ReleaseDate

FROM GameInfo g JOIN ((SELECT GameID

FROM GameInfo

```
WHERE EXTRACT(YEAR FROM ReleaseDate) >= 2000

ORDER BY PlayerEstimate DESC
LIMIT 100)

INTERSECT
(SELECT GameID

FROM GameInfo

ORDER BY Metacritic DESC

LIMIT 100)) as BestGames ON BestGames.GameID = g.GameID

ORDER BY PlayerEstimate DESC, Metacritic DESC, ReleaseDate DESC,
GameName ASC;
```

Top 15 Results for query 2:

```
mysql> SELECT g.GameID, g.GameName, g.PlayerEstimate, g.Metacritic, g.ReleaseDate FROM GameInfo g JOIN ((SELECT GameID FROM GameInfo WHERE EXTRACT
(YEAR FROM ReleaseDate) >= 2000 ORDER BY PlayerEstimate DESC LIMIT 100) INTERSECT (SELECT GameID FROM GameInfo ORDER BY Metacritic DESC LIMIT 100)
) as BestGames ON BestGames.GameID = g.GameID ORDER BY PlayerEstimate DESC, Metacritic DESC, ReleaseDate DESC, GameName ASC LIMIT 15;
+--------+----------------------------+----------------+------------+-------------+
| GameID | GameName                   | PlayerEstimate | Metacritic | ReleaseDate |
+--------+----------------------------+----------------+------------+-------------+
|    570 | Dota 2                     |       90687580 |         90 | 2013-07-09  |
|    440 | Team Fortress 2            |       37878812 |         92 | 2007-10-10  |
|    550 | Left 4 Dead 2              |       13583400 |         89 | 2009-11-16  |
|    240 | Counter-Strike: Source     |       11472993 |         88 | 2004-11-01  |
|  72850 | The Elder Scrolls V: Skyrim |       10903558 |         94 | 2011-11-10  |
|   8930 | Sid Meier's Civilization V |        9150595 |         90 | 2010-09-21  |
|    620 | Portal 2                   |        7282849 |         95 | 2011-04-18  |
|    400 | Portal                     |        6864247 |         90 | 2007-10-10  |
|  49520 | Borderlands 2              |        6263964 |         89 | 2012-09-17  |
| 271590 | Grand Theft Auto V         |        5756584 |         96 | 2015-04-13  |
|    220 | Half-Life 2                |        5695963 |         96 | 2004-11-16  |
|   8870 | BioShock Infinite          |        3526192 |         94 | 2013-03-25  |
|  12210 | Grand Theft Auto IV        |        3520274 |         90 | 2008-12-02  |
|  10500 | Empire: Total War          |        3149917 |         90 | 2009-03-03  |
|    500 | Left 4 Dead                |        3107949 |         89 | 2008-11-17  |
+--------+----------------------------+----------------+------------+-------------+
15 rows in set (0.02 sec)
```

Indexing Optimization for query 2:

Default Indices:

```
---------------------------------------------------------------------------------------------------------------+
| -> Sort: g.PlayerEstimate DESC, g.MetaCritic DESC, g.ReleaseDate DESC, g.GameName  (actual time=27.331..27.335 rows=19 loops=1)
    -> Stream results  (cost=2926.95 rows=100) (actual time=27.078..27.280 rows=19 loops=1)
        -> Nested loop inner join  (cost=2926.95 rows=100) (actual time=27.065..27.216 rows=19 loops=1)
            -> Table scan on BestGames  (cost=2888.24..2891.95 rows=100) (actual time=27.019..27.029 rows=19 loops=1)
                -> Intersect materialize with deduplication  (cost=2888.20..2888.20 rows=100) (actual time=27.016..27.016 rows=100 loops=1)
                    -> Limit: 100 row(s)  (cost=1439.10 rows=100) (actual time=14.760..14.780 rows=100 loops=1)
                        -> Sort: GameInfo.PlayerEstimate DESC, limit input to 100 row(s) per chunk  (cost=1439.10 rows=11106) (actual time=14.758.
.14.770 rows=100 loops=1)
                            -> Filter: (extract(year from GameInfo.ReleaseDate) >= 2000)  (cost=1439.10 rows=11106) (actual time=0.097..12.589 row
s=12430 loops=1)
                                -> Table scan on GameInfo  (cost=1439.10 rows=11106) (actual time=0.092..10.943 rows=13303 loops=1)
                    -> Limit: 100 row(s)  (cost=1439.10 rows=100) (actual time=12.073..12.095 rows=100 loops=1)
                        -> Sort: GameInfo.MetaCritic DESC, limit input to 100 row(s) per chunk  (cost=1439.10 rows=11106) (actual time=12.072..12.
082 rows=100 loops=1)
                            -> Table scan on GameInfo  (cost=1439.10 rows=11106) (actual time=0.045..10.273 rows=13303 loops=1)
            -> Single-row index lookup on g using PRIMARY (GameID=BestGames.GameID)  (cost=0.25 rows=1) (actual time=0.009..0.009 rows=1 loops=19)
 |
+--------------------------------------------------------------------------------------------------------------------------------------
---------------------------------------------------------------------------------------------------------------------------------------
---------------------------------------------------------------------------------------------------------------------------------------
---------------------------------------------------------------------------------------------------------------------------------------
---------------------------------------------------------------------------------------------------------------------------------------
---------------------------------------------------------------------------------------------------------------------------------------
---------------------------------------------------------------------------------------------------------------------------------------
---------------------------------------------------------------------------------------------------------------------------------------
---------------------------------------------------------------------------------------------------------------------------------------
---------------------------------------------------------------------------------------------------------------------------------------
---------------------------------------------------------------------------------------------------------------------------------------
---------------------------------------------------------------------------------------------------------------+
1 row in set (0.03 sec)
```

Costs are very high. Ran in 0.03 sec.

With GameName Index:

```
mysql> CREATE INDEX idx_GameName ON GameInfo(GameName);
Query OK, 0 rows affected (0.27 sec)
Records: 0  Duplicates: 0  Warnings: 0

mysql> SHOW INDEX FROM GameInfo;
+----------+------------+--------------+--------------+-------------+-----------+-------------+----------+--------+------+------------+---------+-
--------------+---------+------------+
| Table    | Non_unique | Key_name     | Seq_in_index | Column_name | Collation | Cardinality | Sub_part | Packed | Null | Index_type | Comment |
Index_comment | Visible | Expression |
+----------+------------+--------------+--------------+-------------+-----------+-------------+----------+--------+------+------------+---------+-
--------------+---------+------------+
| GameInfo |          0 | PRIMARY      |            1 | GameID      | A         |       11106 |     NULL |   NULL |      | BTREE      |         |
              | YES     | NULL       |
| GameInfo |          1 | idx_GameName |            1 | GameName    | A         |       11106 |     NULL |   NULL | YES  | BTREE      |         |
              | YES     | NULL       |
+----------+------------+--------------+--------------+-------------+-----------+-------------+----------+--------+------+------------+---------+-
--------------+---------+------------+
2 rows in set (0.01 sec)
```

```
---------------------------------------------------------------------------------------------------------------+
| -> Sort: g.PlayerEstimate DESC, g.MetaCritic DESC, g.ReleaseDate DESC, g.GameName  (actual time=23.622..23.625 rows=19 loops=1)
    -> Stream results  (cost=2926.95 rows=100) (actual time=23.497..23.593 rows=19 loops=1)
        -> Nested loop inner join  (cost=2926.95 rows=100) (actual time=23.489..23.575 rows=19 loops=1)
            -> Table scan on BestGames  (cost=2888.24..2891.95 rows=100) (actual time=23.462..23.468 rows=19 loops=1)
                -> Intersect materialize with deduplication  (cost=2888.20..2888.20 rows=100) (actual time=23.459..23.459 rows=100 loops=1)
                    -> Limit: 100 row(s)  (cost=1439.10 rows=100) (actual time=13.833..13.854 rows=100 loops=1)
                        -> Sort: GameInfo.PlayerEstimate DESC, limit input to 100 row(s) per chunk  (cost=1439.10 rows=11106) (actual time=13.832.
.13.844 rows=100 loops=1)
                            -> Filter: (extract(year from GameInfo.ReleaseDate) >= 2000)  (cost=1439.10 rows=11106) (actual time=0.049..11.631 row
s=12430 loops=1)
                                -> Table scan on GameInfo  (cost=1439.10 rows=11106) (actual time=0.046..9.917 rows=13303 loops=1)
                    -> Limit: 100 row(s)  (cost=1439.10 rows=100) (actual time=9.488..9.505 rows=100 loops=1)
                        -> Sort: GameInfo.MetaCritic DESC, limit input to 100 row(s) per chunk  (cost=1439.10 rows=11106) (actual time=9.487..9.49
5 rows=100 loops=1)
                            -> Table scan on GameInfo  (cost=1439.10 rows=11106) (actual time=0.031..7.905 rows=13303 loops=1)
            -> Single-row index lookup on g using PRIMARY (GameID=BestGames.GameID)  (cost=0.25 rows=1) (actual time=0.005..0.005 rows=1 loops=19)
 |
+--------------------------------------------------------------------------------------------------------------------------------------
---------------------------------------------------------------------------------------------------------------------------------------
---------------------------------------------------------------------------------------------------------------------------------------
---------------------------------------------------------------------------------------------------------------------------------------
---------------------------------------------------------------------------------------------------------------------------------------
---------------------------------------------------------------------------------------------------------------------------------------
---------------------------------------------------------------------------------------------------------------------------------------
---------------------------------------------------------------------------------------------------------------------------------------
---------------------------------------------------------------------------------------------------------------------------------------
---------------------------------------------------------------------------------------------------------------------------------------
---------------------------------------------------------------------------------------------------------------------------------------
---------------------------------------------------------------------------------------------------------------+
1 row in set (0.03 sec)
```

Costs are comparable to the default. Ran in 0.03 sec, no improvement:

With Metacritic Index:

```
mysql> CREATE INDEX idx_MetaCritic ON GameInfo(MetaCritic);
Query OK, 0 rows affected (0.14 sec)
Records: 0  Duplicates: 0  Warnings: 0

mysql> SHOW INDEX FROM GameInfo;
+----------+------------+---------------+--------------+-------------+-----------+-------------+----------+--------+------+------------+---------
+---------------+---------+------------+
| Table    | Non_unique | Key_name      | Seq_in_index | Column_name | Collation | Cardinality | Sub_part | Packed | Null | Index_type | Comment
| Index_comment | Visible | Expression |
+----------+------------+---------------+--------------+-------------+-----------+-------------+----------+--------+------+------------+---------
+---------------+---------+------------+
| GameInfo |          0 | PRIMARY       |            1 | GameID      | A         |       11106 |     NULL |   NULL |      | BTREE      |
|               | YES     | NULL       |
| GameInfo |          1 | idx_MetaCritic |            1 | MetaCritic  | A         |          72 |     NULL |   NULL | YES  | BTREE      |
|               | YES     | NULL       |
+----------+------------+---------------+--------------+-------------+-----------+-------------+----------+--------+------+------------+---------
+---------------+---------+------------+
2 rows in set (0.01 sec)
```

```
-----------------------------------------------------------------------------------------------------------------+
| -> Sort: g.PlayerEstimate DESC, g.MetaCritic DESC, g.ReleaseDate DESC, g.GameName  (actual time=12.838..12.841 rows=18 loops=1)
    -> Stream results  (cost=1490.90 rows=100) (actual time=12.720..12.811 rows=18 loops=1)
        -> Nested loop inner join  (cost=1490.90 rows=100) (actual time=12.713..12.795 rows=18 loops=1)
            -> Table scan on BestGames  (cost=1452.19..1455.90 rows=100) (actual time=12.693..12.699 rows=18 loops=1)
                -> Intersect materialize with deduplication  (cost=1452.15..1452.15 rows=100) (actual time=12.692..12.692 rows=100 loops=1)
                    -> Limit: 100 row(s)  (cost=1439.10 rows=100) (actual time=12.522..12.542 rows=100 loops=1)
                        -> Sort: GameInfo.PlayerEstimate DESC, limit input to 100 row(s) per chunk  (cost=1439.10 rows=11106) (actual time=12.521.
.12.532 rows=100 loops=1)
                            -> Filter: (extract(year from GameInfo.ReleaseDate) >= 2000)  (cost=1439.10 rows=11106) (actual time=0.047..10.419 row
s=12430 loops=1)
                                -> Table scan on GameInfo  (cost=1439.10 rows=11106) (actual time=0.044..8.841 rows=13303 loops=1)
                    -> Limit: 100 row(s)  (cost=3.05 rows=100) (actual time=0.032..0.072 rows=100 loops=1)
                        -> Covering index scan on GameInfo using idx_MetaCritic (reverse)  (cost=3.05 rows=100) (actual time=0.031..0.062 rows=100
 loops=1)
            -> Single-row index lookup on g using PRIMARY (GameID=BestGames.GameID)  (cost=0.25 rows=1) (actual time=0.005..0.005 rows=1 loops=18)
 |
+-----------------------------------------------------------------------------------------------------------------+
1 row in set (0.01 sec)
```

Costs are much lower than default and GameName. Ran 0.01 sec. Improvement!

With PlayerEstimate Index:

```
mysql> SHOW INDEX FROM GameInfo;
+----------+------------+--------------------+--------------+----------------+-----------+-------------+----------+--------+------+------------+--
-------+---------------+---------+------------+
| Table    | Non_unique | Key_name           | Seq_in_index | Column_name    | Collation | Cardinality | Sub_part | Packed | Null | Index_type | C
omment | Index_comment | Visible | Expression |
+----------+------------+--------------------+--------------+----------------+-----------+-------------+----------+--------+------+------------+--
-------+---------------+---------+------------+
| GameInfo |          0 | PRIMARY            |            1 | GameID         | A         |       11106 |     NULL |   NULL |      | BTREE      |
       |               | YES     | NULL       |
| GameInfo |          1 | idx_PlayerEstimate |            1 | PlayerEstimate | A         |        1915 |     NULL |   NULL | YES  | BTREE      |
       |               | YES     | NULL       |
+----------+------------+--------------------+--------------+----------------+-----------+-------------+----------+--------+------+------------+--
-------+---------------+---------+------------+
2 rows in set (0.01 sec)
```

```
-----------------------------------------------------------------------------------------+
| -> Sort: g.PlayerEstimate DESC, g.MetaCritic DESC, g.ReleaseDate DESC, g.GameName  (actual time=11.018..11.021 rows=19 loops=1)
    -> Stream results  (cost=1490.90 rows=100) (actual time=10.924..10.993 rows=19 loops=1)
        -> Nested loop inner join  (cost=1490.90 rows=100) (actual time=10.917..10.975 rows=19 loops=1)
            -> Table scan on BestGames  (cost=1452.19..1455.90 rows=100) (actual time=10.889..10.895 rows=19 loops=1)
                -> Intersect materialize with deduplication  (cost=1452.15..1452.15 rows=100) (actual time=10.888..10.888 rows=100 loops=1)
                    -> Limit: 100 row(s)  (cost=3.05 rows=100) (actual time=0.166..0.925 rows=100 loops=1)
                        -> Filter: (extract(year from GameInfo.ReleaseDate) >= 2000)  (cost=3.05 rows=100) (actual time=0.165..0.916 rows=100 loop
s=1)
                            -> Index scan on GameInfo using idx_PlayerEstimate (reverse)  (cost=3.05 rows=100) (actual time=0.162..0.899 rows=102
loops=1)
                    -> Limit: 100 row(s)  (cost=1439.10 rows=100) (actual time=9.860..9.877 rows=100 loops=1)
                        -> Sort: GameInfo.MetaCritic DESC, limit input to 100 row(s) per chunk  (cost=1439.10 rows=11106) (actual time=9.860..9.86
7 rows=100 loops=1)
                            -> Table scan on GameInfo  (cost=1439.10 rows=11106) (actual time=0.021..8.242 rows=13303 loops=1)
            -> Single-row index lookup on g using PRIMARY (GameID=BestGames.GameID)  (cost=0.25 rows=1) (actual time=0.004..0.004 rows=1 loops=19)
 |
+-----------------------------------------------------------------------------------------+
-----------------------------------------------------------------------------------------------------------------------------------------------------------
-----------------------------------------------------------------------------------------------------------------------------------------------------------
-----------------------------------------------------------------------------------------------------------------------------------------------------------
-----------------------------------------------------------------------------------------------------------------------------------------------------------
-----------------------------------------------------------------------------------------------------------------------------------------------------------
-----------------------------------------------------------------------------------------------------------------------------------------------------------
-----------------------------------------------------------------------------------------------------------------------------------------------------------
-----------------------------------------------------------------------------------------------------------------------------------------------------------
-----------------------------------------------------------------------------------------+
1 row in set (0.01 sec)
```

Costs are comparable to Metacritic overall. Ran in 0.01 sec. Same as Metacritic index.

Our query sorts by Metacritic score and PlayerEstimate count, which is why indexing on these fields improved our costs and runtime significantly. We decided to move forward with a Metacritic index because this outperformed the default indices and was overall comparable to the PlayerEstimate index. Although the player estimate index also ran in 0.01 seconds and had similar costs, we think we will use the metacritic field more often in our other queries, so having that index will be more useful.