Short Description

We created a game review website that allows personal users to post reviews to a game and commentaries to reviews, as well as business users to release new games and update game information. There are also web administers who can update profiles of all the users.

Final Schema Description

Our final schema is slightly differently from the initial schema.

- PersonalUser table

  For the attribute *Gender*, we changed its type from String to Enum, since a user is either Male or Female.

  For the attribute *Age*, we added a constraint 0<=age<=100 to meet the requirement of marking rubric.

- Game table

  We added a new attribute gameInfo as it would be better to have a brief description for users to know about the game.

  We changed the candidate key from *{gameName, since}* to *{gameName}* and let gameName to be unique, since it's more proper that a company cannot release two games with the same game. Previously, a company could release two games with the same name at different time, which doesn't really make sense.

- Image, Video and Has tables

  We removed these tables because our website currently does not support the operations of them.

- Category table

  We deleted the attribute *totalNum*, since we can count the number of games under a category from other tables.

- Review table

  We deleted the attribute *author* since users may not want others to know what reviews they left.

  The attribute *text* can be NULL so users can quickly leave a short review in the title.

Functional Dependencies

personaluser (country, gender, age, <u>userID</u>, userName, password, mail, notification)

- userID → {userID, userName,password,country, gender, age, mail, notification}

Key implies other attributes.

- userName → {userID, userName, password,country, gender, age, mail, notification}
  No two personal users can have the same username.

busniessuser (<u>userID</u>, userName, password, mail, notificaiton, officialSite)

- userID → {userID, userName, password, notification, mail, officialSite}
  Key implies other attributes.
  userName → {userID, userName, password, notification, mail, officialSite}
  No two business users can have the same username.

webadmin (<u>userID</u>, userName, password, notification, mail)

- userID → {userID, userName, password, notification, mail}
  Key implies other attributes.

- userName → {userID, userName, password, notification, mail}
  No two web administers can have the same username.

game (<u>gameID</u>, gName, userID, since, gameInfo)

- gameID → {gameID, gName, userID, since, gameInfo}
  Key implies other attributes.

- gName → {gameID, gName, since, userID, gameInfo}
  No two games can have the same name.

review (<u>rID</u>, title, text, userID, gameID)

- rID → {rID, title, text, userID, gameID}
  Key implies other attributes.

commentary (<u>cID</u>, text, userID, rID)

- cID → {cID, text, userID, rID}
  Key implies other attributes.

category (<u>cName</u>)

- cName → {cName}
  Key implies other attributes.

thumbup (userID, rID)

- {userID, rID} → {userID, rID}

Key implies other attributes.

belong (gameID, cName)

- {gameID, cName} → {gameID, cName}
  Key implies other attributes.

SQL Queries

Admin

- Create categories
  ```
  SELECT * FROM category
  INSERT INTO category (cName) VALUES (?)
  ```

- Update user profiles
  ```
  SELECT * FROM businessuser
  SELECT * FROM personaluser
  SELECT * FROM businessuser WHERE userID = '".$_SESSION["bid"]."'
  SELECT * FROM personaluser WHERE userID = '".$_SESSION["pid"]."'
  SELECT * FROM businessuser WHERE userName = '".$uname."'
  SELECT * FROM personaluser WHERE userName = '".$uname."'
  SELECT * FROM webadmin WHERE userName = '".$uname."'
  UPDATE businessuser
          SET userName=?, password=?, mail=?, officialSite=?, notification=?
          WHERE userID=?
  UPDATE personaluser
          SET userName=?, password=?, mail=?, gender=?, country=?, notification=?,
          age=?
          WHERE userID=?
  UPDATE webadmin
          SET userName=?, password=?, mail=?, notification=?
          WHERE userID=?
  ```

- Reset uesers' passwords
  ```
  UPDATE personaluser SET password= '666' WHERE userID='".$_POST["personal"]."'
  UPDATE businessuser SET password= '888' WHERE userID='".$_POST["business"]."'
  ```

Business User

- Display the game list
  ```
  SELECT G.gameID, G.gName, R.title, R.time, R.text
          FROM  game G
          INNER JOIN review R ON G.gameID = R.gameID
          WHERE G.userID = "'.$_SESSION['uid'].'"
          ORDER BY gName;
  ```

- Update the profile

```
SELECT * FROM businessuser WHERE userName = '".$uname."'
UPDATE businessuser
        SET userName=?, password=?, mail=?, officialSite=?, notification=?
        WHERE userID=?
```

- Find the top fan

```
SELECT * FROM game WHERE userID="'.$_SESSION["uid"].'"
SELECT p.userID, p.userName
        FROM personaluser p
        WHERE NOT EXISTS
        (SELECT f.gameID FROM
        (SELECT g1.gameID FROM game g1 WHERE g1.userID ="'.$_SESSION["uid"].'") f
        WHERE f.gameID NOT IN
        (SELECT r.gameID FROM review r WHERE r.userID = p.userID))
```

Personal User

- Generate the report

```
SELECT * FROM game g WHERE g.since = (SELECT MAX(since) AS time FROM game)
SELECT * FROM (SELECT avg(f.r_count) as pop, f.userID
        FROM (SELECT g.gameID, g.userID,
        Count(r.rID) AS r_count
        FROM game g
        LEFT OUTER JOIN review r ON g.gameID = r.gameID
        GROUP BY g.gameID) f
        GROUP BY f.userID) z
                WHERE z.pop = (SELECT MAX(v.pop) AS time
                FROM (SELECT avg(f.r_count) as pop, f.userID
        FROM (SELECT g.gameID, g.userID,
        Count(r.rID) AS r_count
        FROM game g
        LEFT OUTER JOIN review r ON g.gameID = r.gameID
        GROUP BY g.gameID) f
        GROUP BY f.userID) v)
SELECT * FROM game g WHERE g.since = (SELECT MIN(since) AS time FROM game)
SELECT * FROM (SELECT avg(f.r_count) as pop, f.userID
        FROM (SELECT g.gameID, g.userID,
        Count(r.rID) AS r_count
        FROM game g
        LEFT OUTER JOIN review r ON g.gameID = r.gameID
        GROUP BY g.gameID) f
        GROUP BY f.userID) z
                WHERE z.pop = (SELECT MIN(v.pop) AS time
                FROM (SELECT avg(f.r_count) as pop, f.userID
        FROM (SELECT g.gameID, g.userID,
        Count(r.rID) AS r_count
        FROM game g
        LEFT OUTER JOIN review r ON g.gameID = r.gameID
        GROUP BY g.gameID) f
        GROUP BY f.userID) v)
SELECT * FROM businessuser WHERE userID ='".$userID."'
```

- Update the profile

```
SELECT * FROM personaluser WHERE userName = '".$uname."'
UPDATE personaluser
        SET userName=?, password=?, mail=?, gender=?, country=?, notification=?,
        age=?
        WHERE userID=?
```

- Browse the game list, click on a game to see the page of that game

```
SELECT b.userName as name, b.officialSite as site, p.pop
        FROM businessuser b, (SELECT avg(f.r_count) as pop, f.userID
        FROM (SELECT g.gameID, g.userID,
        Count(r.rID) AS r_count
        FROM game g
        LEFT OUTER JOIN review r ON g.gameID = r.gameID
        GROUP BY g.gameID) f
        GROUP BY f.userID) p
        WHERE b.userID = p.userID ORDER BY p.pop DESC";
```

- Search for all the reviews edited on a given day

```
SELECT * FROM review WHERE DATE(time) ='".$date."'
```

Game

- Show the game list

```
SELECT gameID,gName,since FROM game ORDER BY since DESC
```

- A user thumps up for a review

```
INSERT INTO thumbup(userID,rID) VALUES( "' .$userID. '","' .$rID. '
```

- Show reviews, commentaries and the number of thumb ups for a game

```
SELECT * FROM game WHERE gameID= $gameID
SELECT * FROM review WHERE gameID= $gameID
SELECT COUNT(userID),rID FROM thumbup WHERE rID = "' .$row['rID']. '" GROUP BY rID
SELECT * FROM commentary WHERE rID = "' .$row['rID']. '"
```

- Personal user: write a commentary to a review

```
SELECT MAX(cID) FROM commentary
INSERT INTO commentary(cID,text,userID,rID)
        VALUES( "' .$cid. '","' .$text. '", "' .$userID. '"  ,"' .$rID. '
```

- Personal user: write a review for a game

```
SELECT MAX(rID) FROM review
INSERT INTO review(rID,title,text,userID,gameID,time) V
        ALUES( "'.$rid.'","'.$title.'", "'.$text.'"  ,"'.$userID.'","'.$gameID.'","'.$time.'")
```

- Sort the game list by either the category or the release date

```
SELECT cName FROM category
SELECT G.gameID,gName,since,gameInfo,userID
        FROM game G,belong B
        WHERE G.gameID=B.gameID AND B.cname = "'.$_SESSION['sortT'].'"
```

```
SELECT B.userName
        FROM businessuser B,game G
        WHERE G.userID=B.userID AND G.userID = "'.$row['userID'].'"
```

Manage reviews/commentaries (personal user)
Manage games (business user)

- See the game information

```
SELECT gName, gameInfo, since FROM game WHERE gameID = '".$_SESSION["gid"]."'
SELECT  cName FROM belong WHERE gameID = '".$_SESSION["gid"]."'
```

- Personal user: edit their reviews

```
UPDATE review SET title=?, text=?, time=? WHERE rID=?
```

- Personal user: post a new review

```
SELECT gameID FROM game WHERE gName='".$gName."'
SELECT MAX(rID) AS rid FROM review
INSERT INTO review (gameID, rID, title, text, time, userID) VALUES (?,?,?,?,?,?)
```

- Personal user: see the reviews/commentaries they posted

```
SELECT r.rID, r.title, r.text, g.gName
        FROM game g, review r
        WHERE r.gameID = g.gameID AND r.rID = '".$_SESSION["rid"]."'
SELECT a.rID, a.title, a.time, a.gameID, a.gName, b.c_count, c.t_count
        FROM (SELECT r.rID, r.title, r.time, g.gameID, g.gName
        FROM game g, review r
        WHERE r.gameID = g.gameID AND r.userID = '".$_SESSION["uid"]."') a,
        FROM (SELECT rID FROM review WHERE userID = '".$_SESSION["uid"]."') r
        LEFT OUTER JOIN commentary c ON c.rID = r.rID
        GROUP BY r.rID) b,
        (SELECT r.rID,
              Count(t.userID) AS t_count
              FROM (SELECT rID FROM review WHERE userID = '".$_SESSION["uid"]."') r
              LEFT OUTER JOIN thumbup t ON t.rID = r.rID
              GROUP BY r.rID) c
              WHERE a.rID = b.rID AND b.rID = c.rID";
SELECT c.cID, c.rID, c.text, r.title
        FROM commentary c, review r
        WHERE c.userID =". $_SESSION["uid]." AND c.rID = r.rID
```

- Businessl user: edit a game

```
UPDATE game SET gName=?, gameInfo=? WHERE gameID=?
INSERT INTO belong (cName, gameID) VALUES ('".$cate."', '".$_SESSION["gid"]."')
DELETE FROM belong WHERE gameID='".$_SESSION["gid"]."' AND cName='".$cate."'
```

- Businessl user: delete a game

```
DELETE FROM game WHERE gameID='".$_POST["gid"]."'
DELETE FROM review WHERE rID='".$_POST["rid"]."'
DELETE FROM commentary WHERE cID='".$_POST["cid"]."'
```

- Businessl user: release a new game

```
SELECT cName FROM category Order by cName
SELECT MAX(gameID) AS gid FROM game
INSERT INTO game (gameID, gName, since, gameInfo, userID) VALUES (?,?,?,?,?)
INSERT INTO belong (cName, gameID) VALUES ('".$cate."', '".$gid."')
```

- Business user: see the games they released

```
SELECT g.gameID, g.gName, g.since,
        Count(r.rID) AS r_count
        FROM (SELECT gameID, gName, since
                FROM game WHERE userID = '".$_SESSION["uid"]."') g
        LEFT OUTER JOIN review r ON g.gameID = r.gameID
        GROUP BY g.gameID";
```

Login and Register

- Login: choose the user type (personal, business or webadmin)

```
SELECT * FROM personaluser WHERE userName = '".$uname."'
SELECT * FROM businessuser WHERE userName = '".$uname."'
SELECT * FROM webadmin WHERE userName = '".$uname."'
```

- Register: personal and business fill in their infomation to register.

```
SELECT * FROM personaluser WHERE userName = '".$uname."'
SELECT userID from personaluser ORDER BY userID DESC LIMIT 1
INSERT INTO personaluser (userID, userName, password, mail, gender, country,
notification, age) VALUES (?, ?, ?, ?, ?, ?, ?, ?)
```

```
SELECT * FROM businessuser WHERE userName = '".$uname."'
SELECT userID from businessuser ORDER BY userID DESC LIMIT 1
INSERT INTO businessuser (userID, userName, password, mail, notification, officialSite)
VALUES (?, ?, ?, ?, ?, ?)
```