```
create table genre
(
  id       int auto_increment
    primary key,
  name     varchar(50)       not null,
  isActive tinyint default 1 not null,
  constraint genre_name_uindex
    unique (name)
);

create table platform
(
  id        int auto_increment
    primary key,
  name      varchar(50)       not null,
  isActive  tinyint default 1 not null,
  imagePath varchar(200)      not null,
  constraint platform_name_uindex
    unique (name)
);

create table publisher
(
  id       int(10) auto_increment
    primary key,
  name     varchar(50)       not null,
  isActive tinyint default 1 not null,
  constraint publisher_name_uindex
    unique (name)
);

create table rating
(
  id       int auto_increment
    primary key,
  name     varchar(6)        not null,
  isActive tinyint default 1 not null,
  constraint rating_name_uindex
    unique (name)
);

create table game
(
  id          int auto_increment
    primary key,
  title       varchar(100)     not null,
  description varchar(700)     null,
  imagePath   varchar(200)     null,
```

```
  publisher_id int             not null,
  genre_id     int             not null,
  rating_id    int             not null,
  currentPrice float           null,
  isActive     tinyint default 1 not null,
  constraint game_genre_id_fk
    foreign key (genre_id) references genre (id),
  constraint game_publisher_id_fk
    foreign key (publisher_id) references publisher (id),
  constraint game_rating_id_fk
    foreign key (rating_id) references rating (id)
);

create table game_platform
(
  game_id       int          not null,
  platform_id   int          not null,
  times_rented  int default 0 not null,
  last_rent_date date         null,
  primary key (game_id, platform_id),
  constraint game_library_game_id_fk
    foreign key (game_id) references game (id),
  constraint game_library_platform_id_fk
    foreign key (platform_id) references platform (id)
);

create table role
(
  id          int auto_increment
    primary key,
  name        varchar(20)  not null,
  description varchar(100) not null,
  constraint Role_name_uindex
    unique (name)
);

create table user
(
  id          int auto_increment
    primary key,
  email       varchar(50) not null,
  passwordHash char(60)    not null,
  role_id     int         not null,
  constraint user_role_id_fk
    foreign key (role_id) references role (id)
);

create table customer
```

```
(
  id      int auto_increment
    primary key,
  name    varchar(20)  not null,
  phone   char(10)     not null,
  card    char(16)     not null,
  address varchar(200) not null,
  user_id int          not null,
  constraint customer_user_userID_fk
    foreign key (user_id) references user (id)
);

create definer = vgrzapp@`%` view GamePlatformInfo as
select `vgrzdb`.`game`.`id`          AS `game_id`,
    `vgrzdb`.`game`.`title`       AS `title`,
    `vgrzdb`.`game`.`imagePath`   AS `imagePath`,
    `vgrzdb`.`game`.`genre_id`    AS `genre_id`,
    `vgrzdb`.`game`.`rating_id`   AS `rating_id`,
    `vgrzdb`.`game`.`publisher_id` AS `publisher_id`,
    `vgrzdb`.`game`.`currentPrice` AS `currentPrice`,
    group_concat(`vgrzdb`.`game_platform`.`platform_id` order by
`vgrzdb`.`game_platform`.`platform_id` ASC separator
              ',')           AS `platform_ids`
from (((((`vgrzdb`.`game_platform` join `vgrzdb`.`game` on ((`vgrzdb`.`game_platform`.`game_id` =
`vgrzdb`.`game`.`id`))) join `vgrzdb`.`genre` on ((`vgrzdb`.`game`.`genre_id` =
`vgrzdb`.`genre`.`id`))) join `vgrzdb`.`rating` on ((`vgrzdb`.`game`.`rating_id` =
`vgrzdb`.`rating`.`id`))) join `vgrzdb`.`publisher` on ((`vgrzdb`.`game`.`publisher_id` =
`vgrzdb`.`publisher`.`id`)))
     join `vgrzdb`.`platform` on ((`vgrzdb`.`game_platform`.`platform_id` =
`vgrzdb`.`platform`.`id`)))
group by `vgrzdb`.`game_platform`.`game_id`;

create definer = vgrzapp@`%` view GamePlatformInfo_Active as
select `vgrzdb`.`game`.`id`          AS `game_id`,
    `vgrzdb`.`game`.`title`       AS `title`,
    `vgrzdb`.`game`.`imagePath`   AS `imagePath`,
    `vgrzdb`.`game`.`genre_id`    AS `genre_id`,
    `vgrzdb`.`game`.`rating_id`   AS `rating_id`,
    `vgrzdb`.`game`.`publisher_id` AS `publisher_id`,
    `vgrzdb`.`game`.`currentPrice` AS `currentPrice`,
    group_concat(`vgrzdb`.`game_platform`.`platform_id` order by
`vgrzdb`.`game_platform`.`platform_id` ASC separator
              ',')           AS `platform_ids`
from (((((`vgrzdb`.`game_platform` join `vgrzdb`.`game` on ((`vgrzdb`.`game_platform`.`game_id` =
`vgrzdb`.`game`.`id`))) join `vgrzdb`.`genre` on ((`vgrzdb`.`game`.`genre_id` =
`vgrzdb`.`genre`.`id`))) join `vgrzdb`.`rating` on ((`vgrzdb`.`game`.`rating_id` =
`vgrzdb`.`rating`.`id`))) join `vgrzdb`.`publisher` on ((`vgrzdb`.`game`.`publisher_id` =
`vgrzdb`.`publisher`.`id`)))
```

```
      join `vgrzdb`.`platform` on ((`vgrzdb`.`game_platform`.`platform_id` =
`vgrzdb`.`platform`.`id`)))
where (`vgrzdb`.`game`.`isActive` and `vgrzdb`.`genre`.`isActive` and `vgrzdb`.`rating`.`isActive`
and
    `vgrzdb`.`publisher`.`isActive` and (`vgrzdb`.`platform`.`isActive` = TRUE))
group by `vgrzdb`.`game_platform`.`game_id`;

create
  definer = vgrzapp@`%` procedure Customer_Create(IN customer_name varchar(20), IN
customer_phone char(10),
                                    IN customer_card char(16), IN customer_address varchar(200),
                                    IN customer_user_id int, OUT msg tinyint(1))
BEGIN
  SET msg = false;
  SET @CustomerExists =
        (SELECT IF((SELECT COUNT(user_id) FROM customer WHERE user_id =
customer_user_id) > 0, true, false));

  IF NOT @CustomerExists THEN
    INSERT INTO customer (name, phone, card, address, user_id)
    VALUES (customer_name, customer_phone, customer_card, customer_address,
customer_user_id);
    SET msg = true;
  end if;
END;

create
  definer = vgrzapp@`%` procedure Customer_Exists_ByUserID(IN customer_user_id int, OUT
msg tinyint(1))
BEGIN
  SET msg = (SELECT IF((SELECT COUNT(user_id) FROM customer WHERE user_id =
customer_user_id LIMIT 1) > 0, true,
                  false));
END;

create
  definer = vgrzapp@`%` procedure Customer_Read_ByUserID(IN customer_user_id int)
BEGIN
  SELECT *
  FROM customer
  WHERE user_id = customer_user_id
  LIMIT 1;
END;

create
  definer = vgrzapp@`%` procedure GamePlatformInfo_Active_Read()
BEGIN
  SELECT *
```

```
  FROM GamePlatformInfo_Active
  ORDER BY title ASC;
END;

create
  definer = vgrzapp@`%` procedure GamePlatform_Create(IN gp_game_id int, IN gp_platform_id
int, OUT msg tinyint(1))
BEGIN
  SET msg = false;
  SET @GamePlatformExists =
        (SELECT IF((SELECT COUNT(game_id) FROM game_platform WHERE game_id =
gp_game_id) > 0, true, false));

  IF NOT @GamePlatformExists THEN
     INSERT INTO game_platform (game_id, platform_id)
     VALUES (gp_game_id, gp_platform_id);
     SET msg = true;
  end if;
END;

create
  definer = vgrzapp@`%` procedure GamePlatform_Read(IN g_id int, IN p_id int)
BEGIN
  SELECT *
  FROM game_platform
  WHERE game_id = g_id
    AND platform_id = p_id;
END;

create
  definer = vgrzapp@`%` procedure GamePlatform_Read_List()
BEGIN
  SELECT *
  FROM game_platform;
END;

create
  definer = vgrzapp@`%` procedure GamePlatform_Update(IN gp_game_id int, IN gp_platform_id
int,
                                      IN add_times_rented int, OUT msg tinyint(1))
BEGIN
  SET msg = false;
  SET @GamePlatformExists =
        (SELECT IF((SELECT COUNT(game_id) FROM game_platform WHERE game_id =
gp_game_id) > 0, true, false));

  IF @GamePlatformExists THEN
     UPDATE game_platform
```

```
      SET platform_id    = gp_platform_id,
        times_rented   = times_rented + add_times_rented,
        last_rent_date = NOW()
      WHERE game_id = gp_game_id;
      SET msg = true;
  END IF;
END;

create
  definer = vgrzapp@`%` procedure Game_Create(IN game_title varchar(50), IN game_description
varchar(700),
                                              IN game_image varchar(200), IN game_publisher_id int,
                                              IN game_genre_id int, IN game_rating_id int, IN game_price float,
                                              OUT msg tinyint(1))
BEGIN
  SET msg = false;
  SET @GameExists = (SELECT IF((SELECT COUNT(title) FROM game WHERE title = game_title) >
0, true, false));

  IF NOT @GameExists THEN
      INSERT INTO game (title, description, imagePath, publisher_id, genre_id, rating_id,
currentPrice)
      VALUES (game_title, game_description, game_image, game_publisher_id, game_genre_id,
game_rating_id, game_price);
      SET msg = true;
  end if;
END;

create
  definer = vgrzapp@`%` procedure Game_Read(IN game_id int)
BEGIN
  SELECT *
  FROM game
  WHERE id = game_id;
END;

create
  definer = vgrzapp@`%` procedure Game_Read_List(IN game_isActive tinyint(1))
BEGIN
  SELECT *
  FROM game
  WHERE isActive = game_isActive
  ORDER BY game.title ASC;
END;

create
  definer = vgrzapp@`%` procedure Game_Update(IN game_id int, IN game_title varchar(50),
                                              IN game_description varchar(700), IN game_image varchar(200),
```

```
                                IN game_price float, IN p_id int, IN g_id int, IN r_id int,
                                IN game_isActive int, OUT msg tinyint(1))
BEGIN
  SET msg = false;
  SET @GameExists = (SELECT IF((SELECT COUNT(id) FROM game WHERE id = game_id) > 0,
true, false));

  IF @GameExists THEN
    UPDATE game
    SET title       = game_title,
      description  = game_description,
      imagePath    = game_image,
      currentPrice = game_price,
      publisher_id = p_id,
      genre_id     = g_id,
      rating_id    = r_id,
      isActive     = game_isActive
    WHERE id = game_id;
    SET msg = true;
  end if;
END;

create
  definer = vgrzapp@`%` procedure Genre_Create(IN genre_name varchar(50), OUT msg tinyint(1))
BEGIN
  SET msg = false;
  SET @GenreExists = (SELECT IF((SELECT COUNT(id) FROM genre WHERE name =
genre_name) > 0, true, false));

  IF NOT @GenreExists THEN
    INSERT INTO genre (name) VALUES (genre_name);
    SET msg = true;
  end if;
END;

create
  definer = vgrzapp@`%` procedure Genre_Read(IN genre_id int)
BEGIN
  SELECT *
  FROM genre
  WHERE id = genre_id;
END;

create
  definer = vgrzapp@`%` procedure Genre_Read_List(IN genre_isActive tinyint(1))
BEGIN
  SELECT *
  FROM genre
```

```
  WHERE isActive = genre_isActive
  ORDER BY genre.name ASC;
END;

create
  definer = vgrzapp@`%` procedure Genre_Update(IN genre_id int, IN genre_name varchar(50),
                                               IN genre_isActive tinyint(1), OUT msg tinyint(1))
BEGIN
  SET msg = false;
  SET @GenreExists = (SELECT IF((SELECT COUNT(id) FROM genre WHERE id = genre_id) > 0,
true, false));

  IF @GenreExists THEN
     UPDATE genre SET name = genre_name, isActive=genre_isActive WHERE id = genre_id;
     SET msg = true;
  end if;
END;

create
  definer = vgrzapp@`%` procedure Platform_Create(IN platform_name varchar(50), IN
platform_imagePath varchar(200),
                                                OUT msg tinyint(1))
BEGIN
  SET msg = false;
  SET @PlatformExists = (SELECT IF((SELECT COUNT(id) FROM platform WHERE name =
platform_name) > 0, true, false));

  IF NOT @PlatformExists THEN
     INSERT INTO platform (name, imagePath) VALUES (platform_name, platform_imagePath);
     SET msg = true;
  end if;
END;

create
  definer = vgrzapp@`%` procedure Platform_Read(IN platform_id int)
BEGIN
  SELECT *
  FROM platform
  WHERE id = platform_id;
END;

create
  definer = vgrzapp@`%` procedure Platform_Read_List(IN platform_isActive tinyint(1))
BEGIN
  SELECT *
  FROM platform
  WHERE isActive = platform_isActive
  ORDER BY name ASC;
```

```
END;

create
  definer = vgrzapp@`%` procedure Platform_Update(IN platform_id int, IN platform_name
varchar(50),
                                                  IN platform_isActive tinyint(1), IN platform_imagePath varchar(200),
                                                  OUT msg tinyint(1))
BEGIN
  SET msg = false;
  SET @PlatformExists = (SELECT IF((SELECT COUNT(id) FROM platform WHERE id =
platform_id) > 0, true, false));

  IF @PlatformExists THEN
     UPDATE platform
     SET name = platform_name, isActive=platform_isActive, imagePath=platform_imagePath
     WHERE id = platform_id;
     SET msg = true;
  end if;
END;

create
  definer = vgrzapp@`%` procedure Publisher_Create(IN publisher_name varchar(50), OUT msg
tinyint(1))
BEGIN
  SET msg = false;
  SET @PublisherExists = (SELECT IF((SELECT COUNT(id) FROM publisher WHERE name =
publisher_name) > 0, true, false));

  IF NOT @PlatformExists THEN
     INSERT INTO publisher (name) VALUES (publisher_name);
     SET msg = true;
  end if;
END;

create
  definer = vgrzapp@`%` procedure Publisher_Read(IN publisher_id int)
BEGIN
  SELECT *
  FROM publisher
  WHERE id = publisher_id;
END;

create
  definer = vgrzapp@`%` procedure Publisher_Read_List(IN publisher_isActive tinyint(1))
BEGIN
  SELECT *
  FROM publisher
  WHERE isActive = publisher_isActive
```

```
  ORDER BY name asc;
END;

create
  definer = vgrzapp@`%` procedure Publisher_Update(IN publisher_id int, IN publisher_name
varchar(50),
                                                   IN publisher_isActive tinyint(1), OUT msg tinyint(1))
BEGIN
  SET msg = false;
  SET @PublisherExists = (SELECT IF((SELECT COUNT(id) FROM publisher WHERE id =
publisher_id) > 0, true, false));

  IF @PublisherExists THEN
     UPDATE publisher SET name = publisher_name, isActive=publisher_isActive WHERE id =
publisher_id;
     SET msg = true;
  end if;
END;

create
  definer = vgrzapp@`%` procedure Rating_Create(IN rating_name varchar(50), OUT msg
tinyint(1))
BEGIN
  SET msg = false;
  SET @RatingExists = (SELECT IF((SELECT COUNT(id) FROM rating WHERE name =
rating_name) > 0, true, false));

  IF NOT @RatingExists THEN
     INSERT INTO rating (name) VALUES (rating_name);
     SET msg = true;
  end if;
END;

create
  definer = vgrzapp@`%` procedure Rating_Read(IN rating_id int)
BEGIN
  SELECT *
  FROM rating
  WHERE id = rating_id;
END;

create
  definer = vgrzapp@`%` procedure Rating_Read_List(IN rating_isActive tinyint(1))
BEGIN
  SELECT *
  FROM rating
  WHERE isActive = rating_isActive
  ORDER BY name ASC;
```

```
END;

create
  definer = vgrzapp@`%` procedure Rating_Update(IN rating_id int, IN rating_name varchar(50),
                                                IN rating_isActive tinyint(1), OUT msg tinyint(1))
BEGIN
  SET msg = false;
  SET @RatingExists = (SELECT IF((SELECT COUNT(id) FROM genre WHERE id = rating_id) > 0,
true, false));

  IF @RatingExists THEN
    UPDATE rating SET name = rating_name, isActive=rating_isActive WHERE id = rating_id;
    SET msg = true;
  end if;
END;

create
  definer = vgrzapp@`%` procedure User_Create(IN user_email varchar(50), IN user_password
char(60),
                                              IN user_role_id int, OUT msg tinyint(1))
BEGIN
  SET msg = false;
  SET @UserExists = (SELECT IF((SELECT COUNT(email) FROM user WHERE email = user_email
LIMIT 1) > 0, true, false));

  IF NOT @UserExists THEN
    INSERT INTO user (email, passwordHash, role_id) VALUES (user_email, user_password,
user_role_id);
    SET msg = true;
  end if;
END;

create
  definer = vgrzapp@`%` procedure User_Exists_ByEmail(IN user_email varchar(50), OUT msg
tinyint(1))
BEGIN
  SET msg = (SELECT IF((SELECT COUNT(email) FROM user WHERE email = user_email LIMIT 1)
> 0, true, false));
END;

create
  definer = vgrzapp@`%` procedure User_Read_ByEmail(IN user_email varchar(50))
BEGIN
  SELECT *
  FROM user
  WHERE email = user_email
  LIMIT 1;
END;
```