JDBC PROJECT

DATABASES

Shubham Jain
A00258743 | GROUP B

Brief

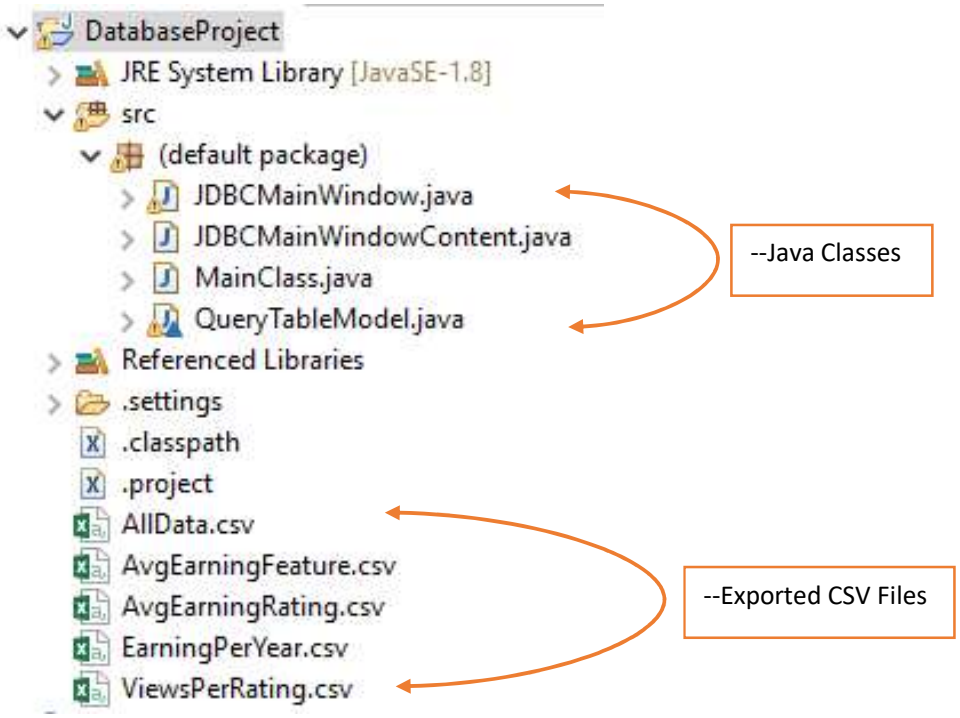
The main aim of this project is to showcase database skills such as JDBC, Stored Procedures, triggers, functions etc.

A Java project template provided has been used and altered to display required data model which has working CRUD operations as well as 4 working filters that export data to excel.

Scope

The scope of this project is limited to mysql queries and functionalities. More work is done on back-end rather than front end.

File Structure



Database Structure

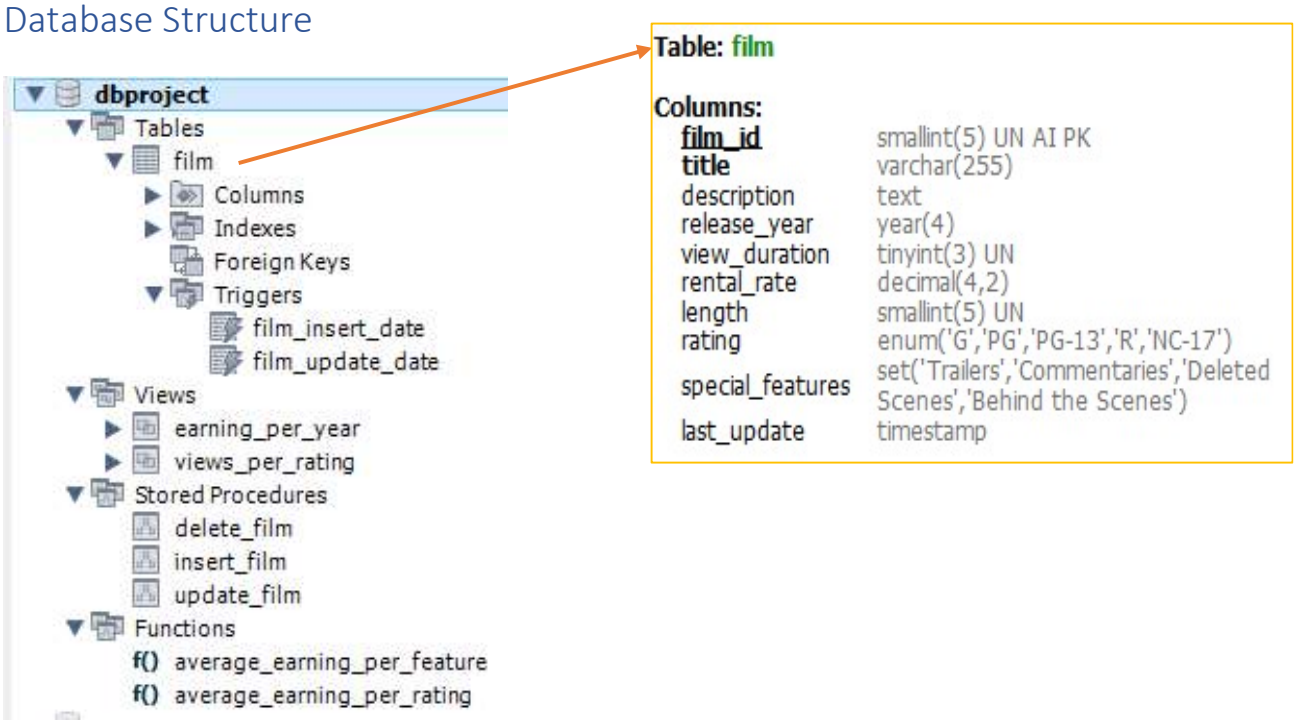


TABLE FEATURES

- MOST SUITABLE DATATYPES WITH DEFAULT VALUES ARE USED
- PRIMARY KEY IS AUTO-INCREMENTED
- LAST_UPDATE COLUMN HAS A TRIGGER FOR INSERTING/UPDATING VALUES

SQL

```
DROP TABLE IF EXISTS `film`;
/*!40101 SET @saved_cs_client     = @@character_set_client */;
 SET character_set_client = utf8mb4 ;
CREATE TABLE `film` (
  `film_id` smallint(5) unsigned NOT NULL AUTO_INCREMENT,
  `title` varchar(255) NOT NULL,
  `description` text,
  `release_year` year(4) DEFAULT NULL,
  `view_duration` tinyint(3) unsigned NOT NULL DEFAULT '3',
  `rental_rate` decimal(4,2) NOT NULL DEFAULT '4.99',
  `length` smallint(5) unsigned DEFAULT NULL,
  `rating` enum('G','PG','PG-13','R','NC-17') DEFAULT 'G',
  `special_features` set('Trailers','Commentaries','Deleted Scenes','Behind the Scenes') DEFAULT NULL,
  `last_update` timestamp NOT NULL DEFAULT CURRENT_TIMESTAMP ON UPDATE CURRENT_TIMESTAMP,
  PRIMARY KEY (`film_id`),
  KEY `idx_title` (`title`)
) ENGINE=InnoDB AUTO_INCREMENT=28 DEFAULT CHARSET=utf8;
/*!40101 SET character_set_client = @saved_cs_client */;
```

TRIGGERS

- Triggers are used to update a column whenever Insert/Update is executed.
- It updates current time in the column.

SQL

Before Insert

```
CREATE DEFINER=`root`@`%` TRIGGER `film_insert_date`
BEFORE INSERT ON `film` FOR EACH ROW
SET NEW.last_update = NOW()
```

Before Update

```
CREATE DEFINER=`root`@`%` TRIGGER `film_update_date`
BEFORE UPDATE ON `film` FOR EACH ROW
SET NEW.last_update = NOW();
```

PROCEDURES

- Stored Procedures are used in this project from Insert, Update and Delete operations.

SQL

Insert

```
DELIMITER $$
CREATE DEFINER=`root`@`%` PROCEDURE `insert_film`(
  p_title    VARCHAR(255),p_description text, p_release_year year(4), p_view_duration tinyint(3),
  p_rental_rate decimal(4,2), p_length smallint(5),p_rating enum('G','PG','PG-13','R','NC-17'),
  p_special_features set('Trailers','Commentaries','Deleted Scenes','Behind the Scenes')
)
BEGIN
  DECLARE sql_error TINYINT DEFAULT FALSE;

  DECLARE CONTINUE HANDLER FOR SQLEXCEPTION
    SET sql_error = TRUE;
    START TRANSACTION;
    INSERT INTO `dbproject`.`film`(`title`,`description`,`release_year`,`view_duration`,`rental_rate`,`length`,`rating`,`special_features`)
    VALUES(p_title ,p_description,p_release_year ,p_view_duration,p_rental_rate,p_length,p_rating,p_special_features);

IF sql_error = FALSE THEN
    COMMIT;
  ELSE
    ROLLBACK;
  END IF;
END$$
DELIMITER ;
```

UPDATE

```
DELIMITER $$
CREATE DEFINER=`root`@`%` PROCEDURE `update_film`(
p_id smallint(5),p_title    VARCHAR(255),p_description text,p_release_year year(4),p_view_duration tinyint(3),
p_rental_rate decimal(4,2),p_length smallint(5),p_rating enum('G','PG','PG-13','R','NC-17'),
p_special_features set('Trailers','Commentaries','Deleted Scenes','Behind the Scenes')
)
BEGIN
  DECLARE sql_error TINYINT DEFAULT FALSE;
  DECLARE CONTINUE HANDLER FOR SQLEXCEPTION
    SET sql_error = TRUE;
    START TRANSACTION;
    UPDATE `dbproject`.`film` SET `title` = p_title, `description` = p_description,`release_year` = p_release_year,
    `view_duration` = p_view_duration, `rental_rate` = p_rental_rate, `length` = p_length, `rating` = p_rating,
    `special_features` = p_special_features

WHERE `film_id` = p_id;
IF sql_error = FALSE THEN
    COMMIT;
  ELSE
    ROLLBACK;
  END IF;
END$$
DELIMITER ;
```

DELETE

```
DELIMITER $$
CREATE DEFINER=`root`@`%` PROCEDURE `delete_film`(
   p_id smallint(5)
)
BEGIN
  DECLARE sql_error TINYINT DEFAULT FALSE;

  DECLARE CONTINUE HANDLER FOR SQLEXCEPTION
    SET sql_error = TRUE;
   START TRANSACTION;
DELETE FROM `dbproject`.`film`
WHERE film_id=p_id;

IF sql_error = FALSE THEN
    COMMIT;
  ELSE
    ROLLBACK;
  END IF;
END$$
DELIMITER ;
```

VIEWS

- VIEWS HAVE BEEN USED TO EXPORT DATA WHICH HAVE QUERIES WITH NO PARAMETERS

SQL

1. EARNING PER YEAR

```
CREATE ALGORITHM=UNDEFINED DEFINER=`root`@`%` SQL SECURITY
DEFINER VIEW `earning_per_year` AS
select
(`film`.`view_duration` * `film`.`rental_rate`) AS `Income`,
`film`.`release_year`
from `film`
group by `film`.`release_year`
order by `Income` desc;
```

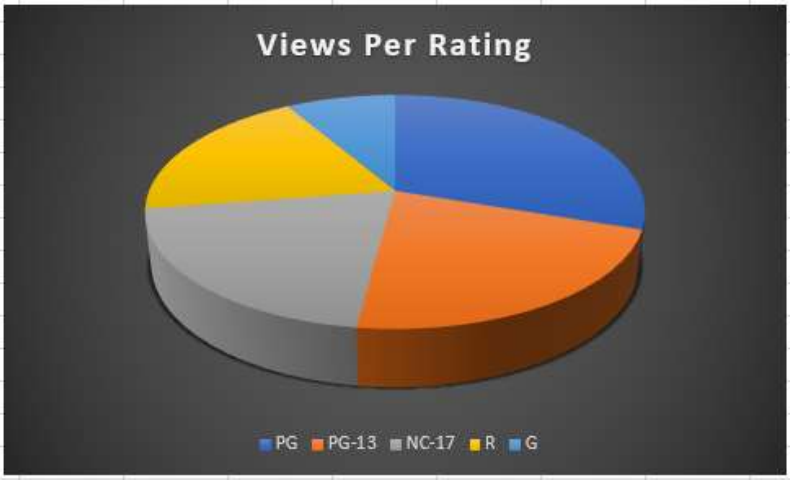
EXCEL OUTPUT

2. Views Per Rating Type

```
CREATE ALGORITHM=UNDEFINED DEFINER=`root`@`%` SQL SECURITY
DEFINER VIEW `views_per_rating` AS
select (`film`.`view_duration` * `film`.`length`) AS `View_Minutes`,
`film`.`rating` AS `rating` from `film`
group by `film`.`rating`
order by `View_Minutes` desc;
```

Excel OutPut

View_Minutes	rating
516	PG
372	PG-13
350	NC-17
324	R
144	G



FUNCTIONS

- FUNCTIONS HAVE BEEN USED TO EXPORT DATA FOR QUERIES WITH PARAMETERS

SQL

1. AVERAGE EARNING RATING WISE

```
DELIMITER $$
CREATE DEFINER=`root`@`%` FUNCTION `average_earning_per_rating`(
rating varchar(25)
) RETURNS varchar(25) CHARSET utf8mb4
BEGIN
  DECLARE Income  varchar(25);
    select avg(rental_rate*length)
    into Income
    from film
    where rating = rating;
 RETURN Income;
END$$
DELIMITER ;
```

	A	B	C	D
1	average_earning_per_rating("PG")			
2	234.2779			

2. AVERAGE EARNING FEATURE WISE

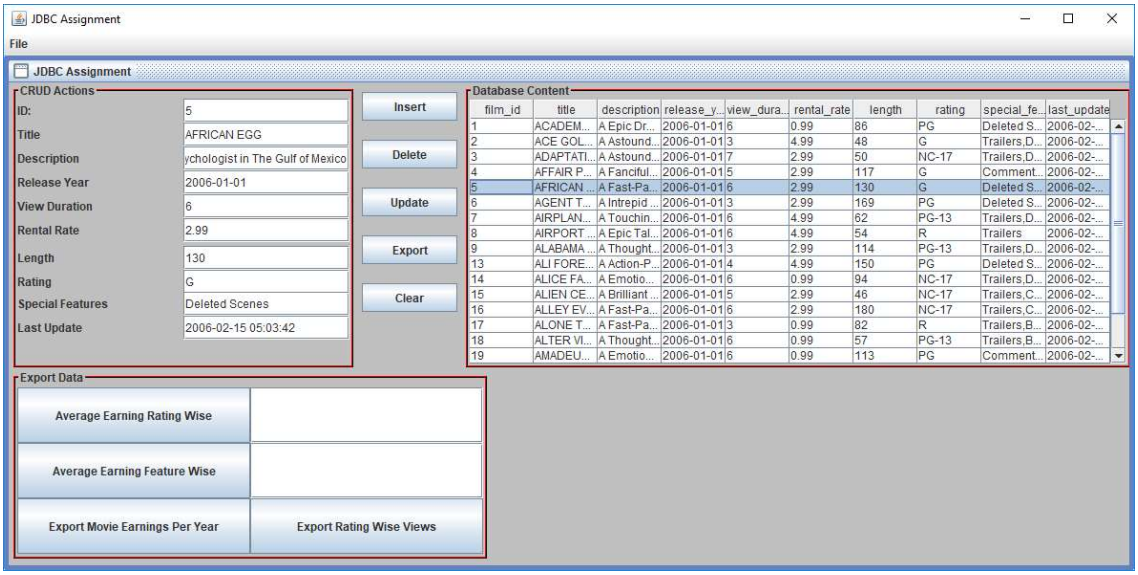
```
DELIMITER $$
CREATE DEFINER=`root`@`%` FUNCTION `average_earning_per_feature`(
feature varchar(25)
) RETURNS varchar(25) CHARSET utf8mb4
BEGIN
  DECLARE Income  varchar(25);

    select avg(rental_rate*length)
    into Income
    from film
    where special_features like CONCAT('%', feature, '%');
 RETURN Income;
END$$
DELIMITER ;
```

	A	B	C	D
1	average_earning_per_feature("Trailers")			
2	173.9946			

GUI

On Clicling a row from Jtable all values get automatically populated, ready for CRUD operations.



Conclusion

Thus this project contains all the back-end featured which are advanced and required for this JDBC project.