JAIME RUIZ 1ºDAM

Final Report: Sakila Database Management in MySQL Workbench

1. Tools and SQL Statements for Modifying Database Content

To work with the database, I used different SQL commands:

- INSERT to add new data.
- UPDATE to modify existing records.
- DELETE to remove records.
- ALTER to change table structures.

Functionality of the tools in MySQL Workbench:

- SQL Editor to run queries.
- Schema Inspector to explore database structures.
- Query Builder to create complex queries visually.

2. Data Insertion, Deletion, and Update

For this part, I performed the following tasks on the actor table:

- Added a new actor with fictitious details.
- Updated the last name of an existing actor.
- Deleted an actor from the table.

Each operation was executed successfully, ensuring changes were reflected in the database.

Results:

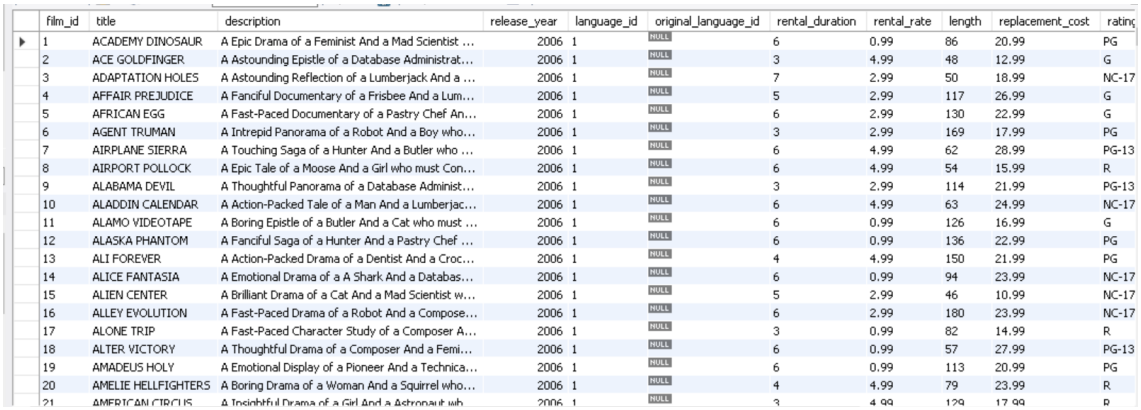
actor_id	first_name	last_name	last_update
192	JOHN	SUVARI	2025-03-10 13:27:11
201	Jaime	Ruiz	2025-03-10 13:31:39
NULL	NULL	NULL	NULL

actor_id	first_name	last_name	last_update
192	JOHN	SMITH	2025-03-10 13:34:16
NULL	NULL	NULL	NULL

3. Creating a Table from a Query Result

To analyze films released after 2005, I extracted relevant records from the film table and stored them in a new table called recent_films. This allows for easier querying and organization of recent movie data.

Results:



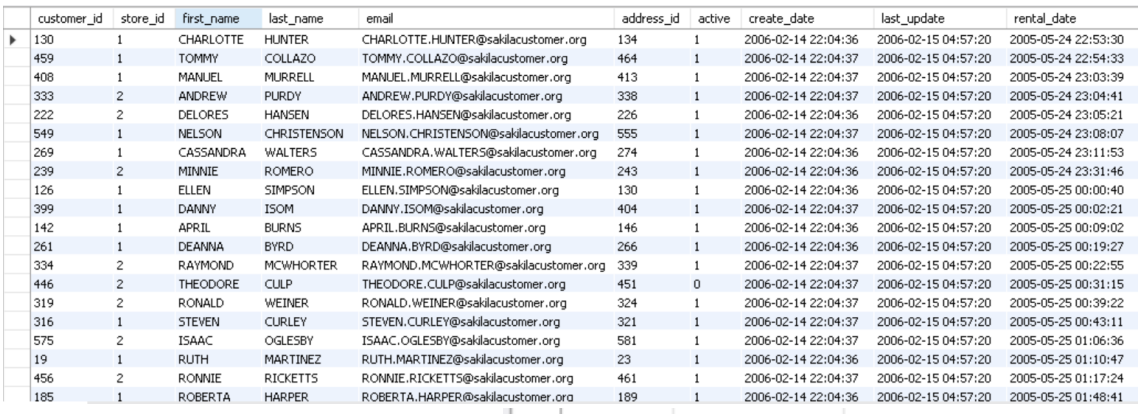
film_id	title	description	release_year	language_id	original_language_id	rental_duration	rental_rate	length	replacement_cost	rating
1	ACADEMY DINOSAUR	A Epic Drama of a Feminist And a Mad Scientist ...	2006	1	NULL	6	0.99	86	20.99	PG
2	ACE GOLDFINGER	A Astounding Epistle of a Database Administrat...	2006	1	NULL	3	4.99	48	12.99	G
3	ADAPTATION HOLES	A Astounding Reflection of a Lumberjack And a ...	2006	1	NULL	7	2.99	50	18.99	NC-17
4	AFFAIR PREJUDICE	A Fanciful Documentary of a Frisbee And a Lum...	2006	1	NULL	5	2.99	117	26.99	G
5	AFRICAN EGG	A Fast-Paced Documentary of a Pastry Chef An...	2006	1	NULL	6	2.99	130	22.99	G
6	AGENT TRUMAN	A Intrepid Panorama of a Robot And a Boy who...	2006	1	NULL	3	2.99	169	17.99	PG
7	AIRPLANE SIERRA	A Touching Saga of a Hunter And a Butler who ...	2006	1	NULL	6	4.99	62	28.99	PG-13
8	AIRPORT POLLOCK	A Epic Tale of a Moose And a Girl who must Con...	2006	1	NULL	6	4.99	54	15.99	R
9	ALABAMA DEVIL	A Thoughtful Panorama of a Database Administ...	2006	1	NULL	3	2.99	114	21.99	PG-13
10	ALADDIN CALENDAR	A Action-Packed Tale of a Man And a Lumberjac...	2006	1	NULL	6	4.99	63	24.99	NC-17
11	ALAMO VIDEOTAPE	A Boring Epistle of a Butler And a Cat who must ...	2006	1	NULL	6	0.99	126	16.99	G
12	ALASKA PHANTOM	A Fanciful Saga of a Hunter And a Pastry Chef ...	2006	1	NULL	6	0.99	136	22.99	PG
13	ALI FOREVER	A Action-Packed Drama of a Dentist And a Croc...	2006	1	NULL	4	4.99	150	21.99	PG
14	ALICE FANTASIA	A Emotional Drama of a A Shark And a Databas...	2006	1	NULL	6	0.99	94	23.99	NC-17
15	ALIEN CENTER	A Brilliant Drama of a Cat And a Mad Scientist w...	2006	1	NULL	5	2.99	46	10.99	NC-17
16	ALLEY EVOLUTION	A Fast-Paced Drama of a Robot And a Compose...	2006	1	NULL	6	2.99	180	23.99	NC-17
17	ALONE TRIP	A Fast-Paced Character Study of a Composer A...	2006	1	NULL	3	0.99	82	14.99	R
18	ALTER VICTORY	A Thoughtful Drama of a Composer And a Femi...	2006	1	NULL	6	0.99	57	27.99	PG-13
19	AMADEUS HOLY	A Emotional Display of a Pioneer And a Technica...	2006	1	NULL	6	0.99	113	20.99	PG
20	AMELIE HELLFIGHTERS	A Boring Drama of a Woman And a Squirrel who...	2006	1	NULL	4	4.99	79	23.99	R
21	AMERICAN CIRCUS	A Insightful Drama of a Girl And a Astronaut wh...	2006	1	NULL	3	4.99	129	17.99	R

4. Complex SQL Queries

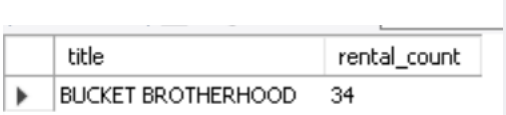
I wrote queries to:

- Retrieve customers who rented a movie in the last 30 days.
- Identify the most rented film in the database.
- Calculate the total revenue generated by each store.

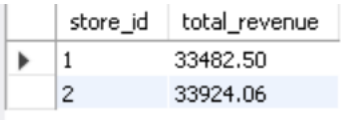
Results:



customer_id	store_id	first_name	last_name	email	address_id	active	create_date	last_update	rental_date
130	1	CHARLOTTE	HUNTER	CHARLOTTE.HUNTER@sakilacustomer.org	134	1	2006-02-14 22:04:36	2006-02-15 04:57:20	2005-05-24 22:53:30
459	1	TOMMY	COLLAZO	TOMMY.COLLAZO@sakilacustomer.org	464	1	2006-02-14 22:04:37	2006-02-15 04:57:20	2005-05-24 22:54:33
408	1	MANUEL	MURRELL	MANUEL.MURRELL@sakilacustomer.org	413	1	2006-02-14 22:04:37	2006-02-15 04:57:20	2005-05-24 23:03:39
333	2	ANDREW	PURDY	ANDREW.PURDY@sakilacustomer.org	338	1	2006-02-14 22:04:37	2006-02-15 04:57:20	2005-05-24 23:04:41
222	2	DELORES	HANSEN	DELORES.HANSEN@sakilacustomer.org	226	1	2006-02-14 22:04:36	2006-02-15 04:57:20	2005-05-24 23:05:21
549	1	NELSON	CHRISTENSON	NELSON.CHRISTENSON@sakilacustomer.org	555	1	2006-02-14 22:04:37	2006-02-15 04:57:20	2005-05-24 23:08:07
269	1	CASSANDRA	WALTERS	CASSANDRA.WALTERS@sakilacustomer.org	274	1	2006-02-14 22:04:36	2006-02-15 04:57:20	2005-05-24 23:11:53
239	2	MINNIE	ROMERO	MINNIE.ROMERO@sakilacustomer.org	243	1	2006-02-14 22:04:36	2006-02-15 04:57:20	2005-05-24 23:31:46
126	1	ELLEN	SIMPSON	ELLEN.SIMPSON@sakilacustomer.org	130	1	2006-02-14 22:04:36	2006-02-15 04:57:20	2005-05-25 00:00:40
399	1	DANNY	ISOM	DANNY.ISOM@sakilacustomer.org	404	1	2006-02-14 22:04:37	2006-02-15 04:57:20	2005-05-25 00:02:21
142	1	APRIL	BURNS	APRIL.BURNS@sakilacustomer.org	146	1	2006-02-14 22:04:36	2006-02-15 04:57:20	2005-05-25 00:09:02
261	1	DEANNA	BYRD	DEANNA.BYRD@sakilacustomer.org	266	1	2006-02-14 22:04:36	2006-02-15 04:57:20	2005-05-25 00:19:27
334	2	RAYMOND	MCWHORTER	RAYMOND.MCWHORTER@sakilacustomer.org	339	1	2006-02-14 22:04:37	2006-02-15 04:57:20	2005-05-25 00:22:55
446	2	THEODORE	CULP	THEODORE.CULP@sakilacustomer.org	451	0	2006-02-14 22:04:37	2006-02-15 04:57:20	2005-05-25 00:31:15
319	2	RONALD	WEINER	RONALD.WEINER@sakilacustomer.org	324	1	2006-02-14 22:04:37	2006-02-15 04:57:20	2005-05-25 00:39:22
316	1	STEVEN	CURLEY	STEVEN.CURLEY@sakilacustomer.org	321	1	2006-02-14 22:04:37	2006-02-15 04:57:20	2005-05-25 00:43:11
575	2	ISAAC	OGLESBY	ISAAC.OGLESBY@sakilacustomer.org	581	1	2006-02-14 22:04:37	2006-02-15 04:57:20	2005-05-25 01:06:36
19	1	RUTH	MARTINEZ	RUTH.MARTINEZ@sakilacustomer.org	23	1	2006-02-14 22:04:36	2006-02-15 04:57:20	2005-05-25 01:10:47
456	2	RONNIE	RICKETTS	RONNIE.RICKETTS@sakilacustomer.org	461	1	2006-02-14 22:04:37	2006-02-15 04:57:20	2005-05-25 01:17:24
185	1	ROBERTA	HARPER	ROBERTA.HARPER@sakilacustomer.org	189	1	2006-02-14 22:04:36	2006-02-15 04:57:20	2005-05-25 01:48:41



title	rental_count
BUCKET BROTHERHOOD	34



store_id	total_revenue
1	33482.50
2	33924.06

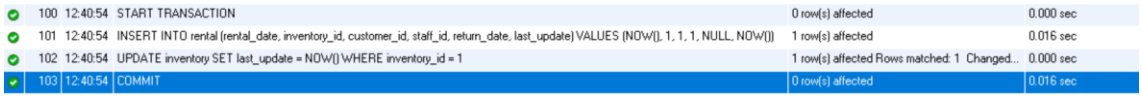
5. Understanding Transactions

Transactions are crucial for maintaining data consistency. To understand how they work, I performed a transaction that:

- Inserted a new rental record.
- Updated the inventory to reflect the rental.
- Committed the changes to ensure database integrity.

This ensures that all steps are completed successfully before finalizing changes.

Results:



✓	100	12:40:54	START TRANSACTION	0 row(s) affected	0.000 sec
✓	101	12:40:54	INSERT INTO rental (rental_date, inventory_id, customer_id, staff_id, return_date, last_update) VALUES (NOW(), 1, 1, 1, NULL, NOW())	1 row(s) affected	0.016 sec
✓	102	12:40:54	UPDATE inventory SET last_update = NOW() WHERE inventory_id = 1	1 row(s) affected Rows matched: 1 Changed...	0.000 sec
✓	103	12:40:54	COMMIT	0 row(s) affected	0.016 sec

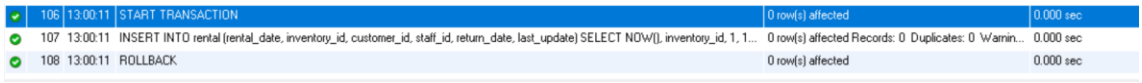
6. Rolling Back Transactions

A rollback is essential when a transaction cannot be completed due to an issue, such as renting a movie that is out of stock. To demonstrate this, I:

- Began a transaction.
- Attempted to insert a rental record for a movie that was unavailable.
- Rolled back the transaction, ensuring no incorrect data was saved.

This prevents incomplete or incorrect entries in the database.

Results:



✓	106	13:00:11	START TRANSACTION	0 row(s) affected	0.000 sec
✓	107	13:00:11	INSERT INTO rental (rental_date, inventory_id, customer_id, staff_id, return_date, last_update) SELECT NOW(), inventory_id, 1, 1...	0 row(s) affected Records: 0 Duplicates: 0 Warnin...	0.000 sec
✓	108	13:00:11	ROLLBACK	0 row(s) affected	0.000 sec

7. Record Locking Policies

To understand how MySQL handles concurrent data modifications, I examined:

- Pessimistic Locking: Prevents other users from modifying a record until the current transaction is complete.
- Optimistic Locking: Allows multiple transactions but checks for conflicts before committing changes.

I simulated a scenario where two users attempted to update the same record simultaneously and analysed how MySQL handled the conflict.

Results:

USER 1 AND 2 CONSOLE (STEP 1):

```
mysql> USE sakila;
Database changed
mysql> SELECT customer_id, first_name, last_name FROM customer LIMIT 1;
+-------------+------------+-----------+
| customer_id | first_name | last_name |
+-------------+------------+-----------+
|           1 | MARY       | SMITH     |
+-------------+------------+-----------+
1 row in set (0.00 sec)
```

USER 1 CONSOLE (STEP 2):

```
mysql> START TRANSACTION;
Query OK, 0 rows affected (0.00 sec)

mysql> SELECT * FROM customer WHERE customer_id = 5 FOR UPDATE;
+-------------+----------+------------+-----------+-----------------------------------+------------+--------+---------------------+---------------------+
| customer_id | store_id | first_name | last_name | email                             | address_id | active | create_date         | last_update         |
+-------------+----------+------------+-----------+-----------------------------------+------------+--------+---------------------+---------------------+
|           5 |        1 | ELIZABETH  | BROWN     | ELIZABETH.BROWN@sakilacustomer.org |          9 |      1 | 2006-02-14 22:04:36 | 2006-02-15 04:57:20 |
+-------------+----------+------------+-----------+-----------------------------------+------------+--------+---------------------+---------------------+
1 row in set (0.00 sec)

mysql> UPDATE customer SET last_name = 'SMITH' WHERE customer_id = 5;
Query OK, 1 row affected (0.00 sec)
Rows matched: 1  Changed: 1  Warnings: 0

mysql> -- DO NOT COMMIT YET!
```

USER 2 CONSOLE (STEP 2):

```
mysql> START TRANSACTION;
Query OK, 0 rows affected (0.00 sec)

mysql> UPDATE customer SET last_name = 'JOHNSON' WHERE customer_id = 5;

mysql> -- This query will HANG until Session 1 commits or rolls back
```

IF USER 1 COMMITS:

```
mysql> COMMIT;
Query OK, 0 rows affected (0.01 sec)
```

IF USER 1 ROLLBACKS:

```
mysql> ROLLBACK;
Query OK, 0 rows affected (0.00 sec)
```

8. Ensuring Data Integrity and Consistency

Maintaining data integrity is crucial in any database system. I identified potential issues in the Sakila database, including:

- Orphaned records due to missing foreign keys.
- Data inconsistencies from a lack of constraints.
- Data duplication due to missing uniqueness constraints.

To address these issues, I implemented:

- Foreign Key Constraints to enforce relationships between tables and prevent orphaned records.
- Triggers to validate and control modifications, ensuring only valid data is inserted or updated.

These measures help maintain a clean and reliable database.

**(MySQL workbench was always crashing when running these queries)*