Sakila
Sample Database

Sang Shin
www.javapassion.com
“Learning is fun!”

Topics

- What is Sakila sample database?
- film inventory related tables
- customer related tables
- rental business related tables
- Views
- Stored routines
- Triggers
- Examples

What is Sakila Sample Database?

Sakila Sample Database

- Tables represent DVD rental store
- Types of tables
  - film inventory related tables
  - customer related tables
  - rental business related tables
- Various MySQL features are used
  - Views
  - Triggers
  - Stored routines

Film Inventory Tables

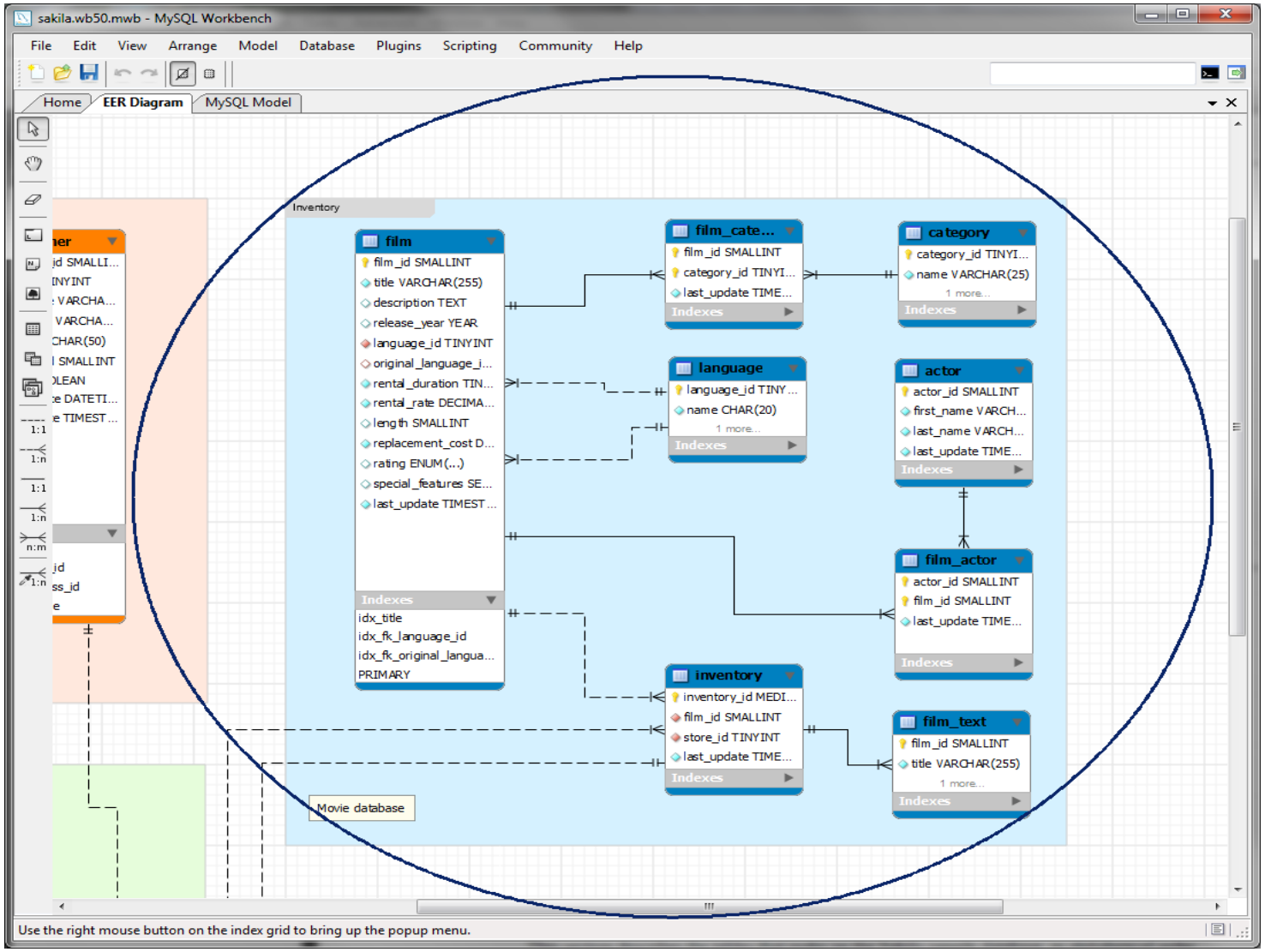
Film Inventory Tables

- film
- category
- film_category
  - many-to-many join table between film and category
- actor
- film_actor
  - many-to-many join table between film and actor
- language

Film Inventory Tables

- Inventory
  - One row for each copy of a given film in a store
  - Refers to film and store tables through foreign keys
- film_text
  - Only table that uses MyISAM storage engine
  - Allows for fulltext searching of the titles and descriptions of the films in the film table
  - Updated via triggers - whenever INSERT, UPDATE, DELETE events occur on film table, trigger update film_text table

Film Inventory Tables



Customer Related Tables

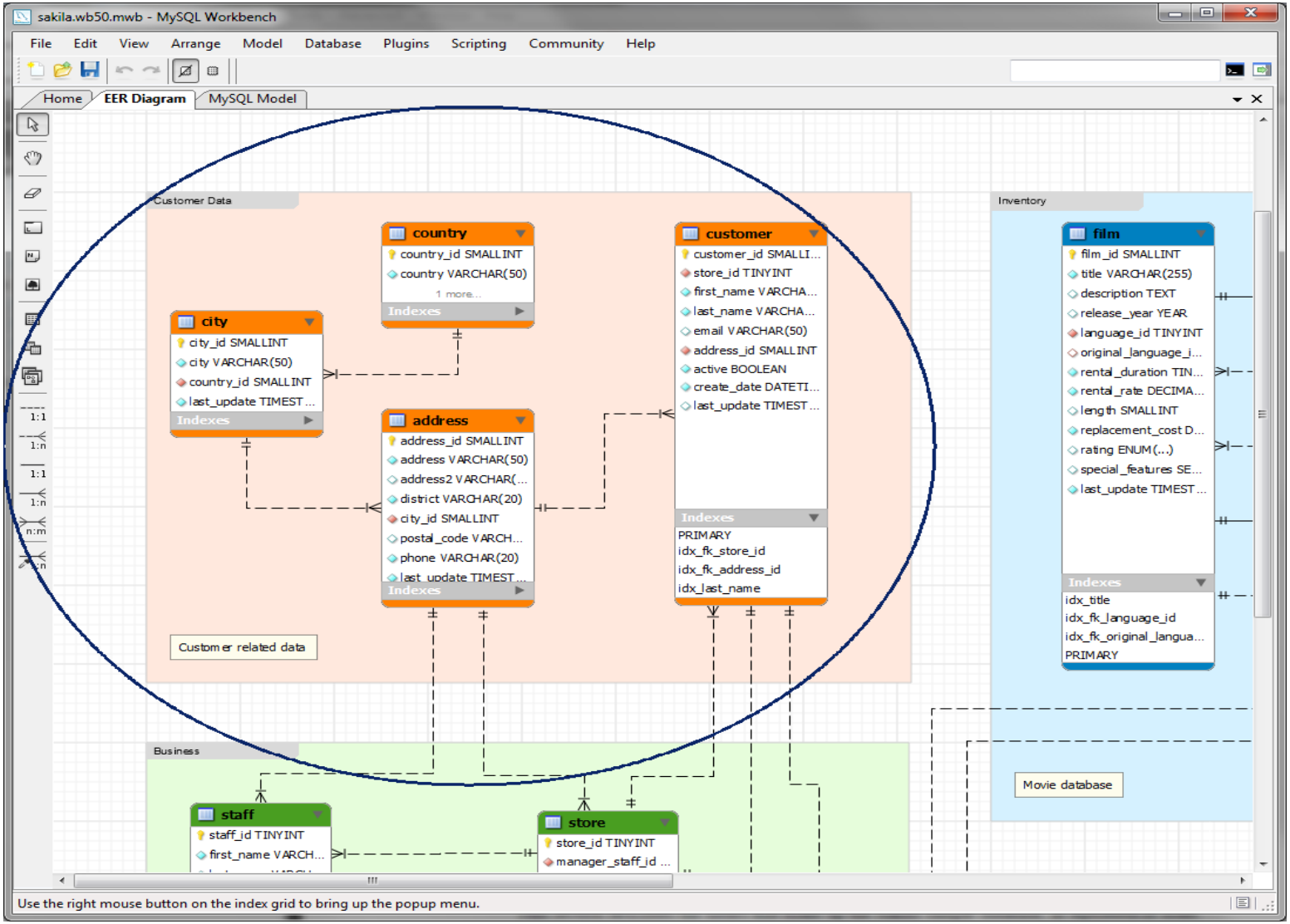
Customer Related Tables

- customer
  - List of all customers
  - Referred to in the payment and rental tables
  - Refers to address and store table via foreign keys
- city
  - List of cities
  - Referred to in the address table
  - Refers to country table via foreign key

Customer Related Tables

- country
  - List of all countries
  - Referred to in the city table
- address
  - List of addresses of customers, staff, and stores
  - Referred to in the customer, staff, and store tables

Customer-related Tables



Rental Business Related Tables

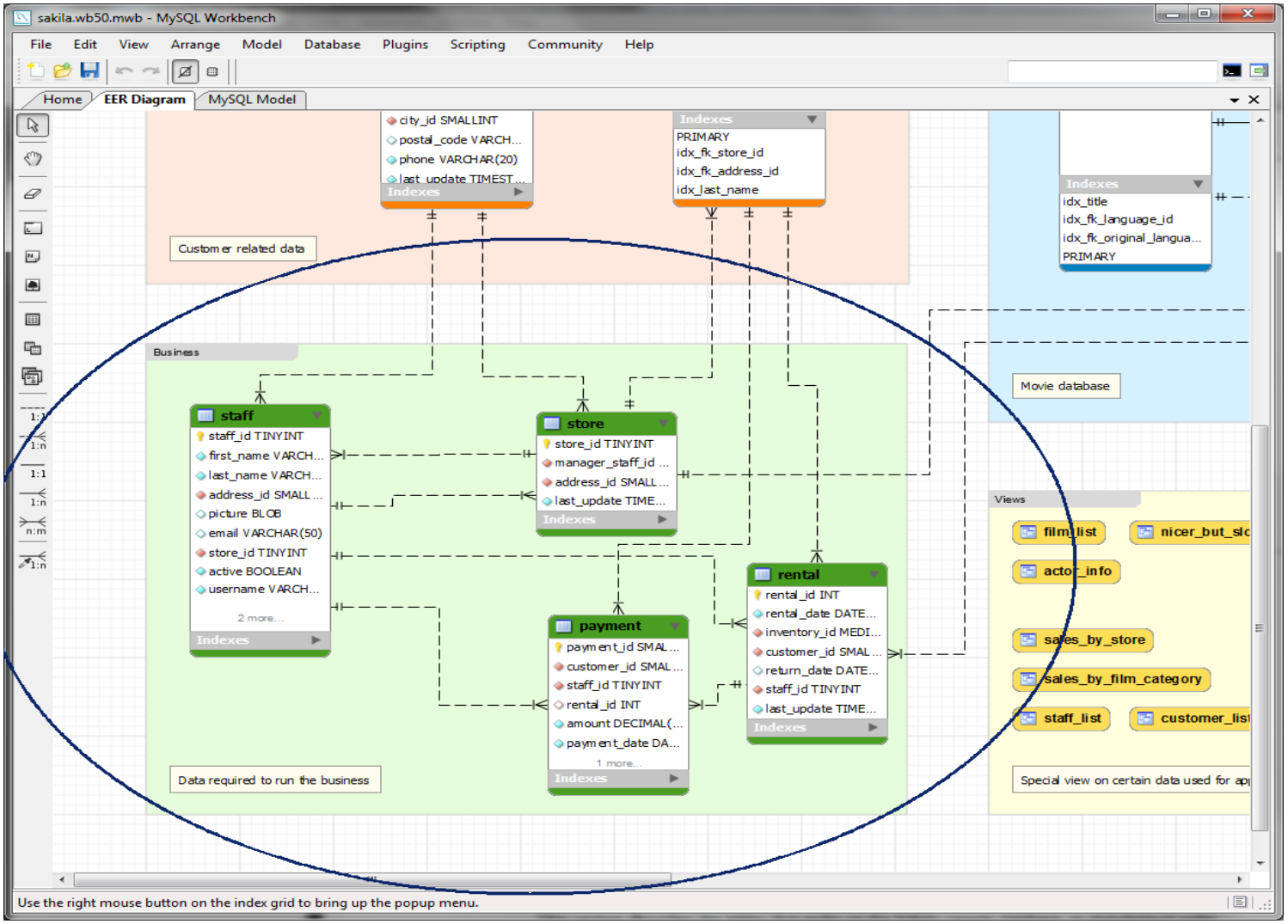
Rental business related tables

- store
  - List of all stores
  - All inventory is assigned to a specific store
  - Staff and customers are assigned to a “home” store
  - Referred to in the staff, customer, and inventory tables
- staff
  - List of all staff members
  - Referred to in the rental, payment, and store tables
  - Refers to store and address tables

Rental business related tables

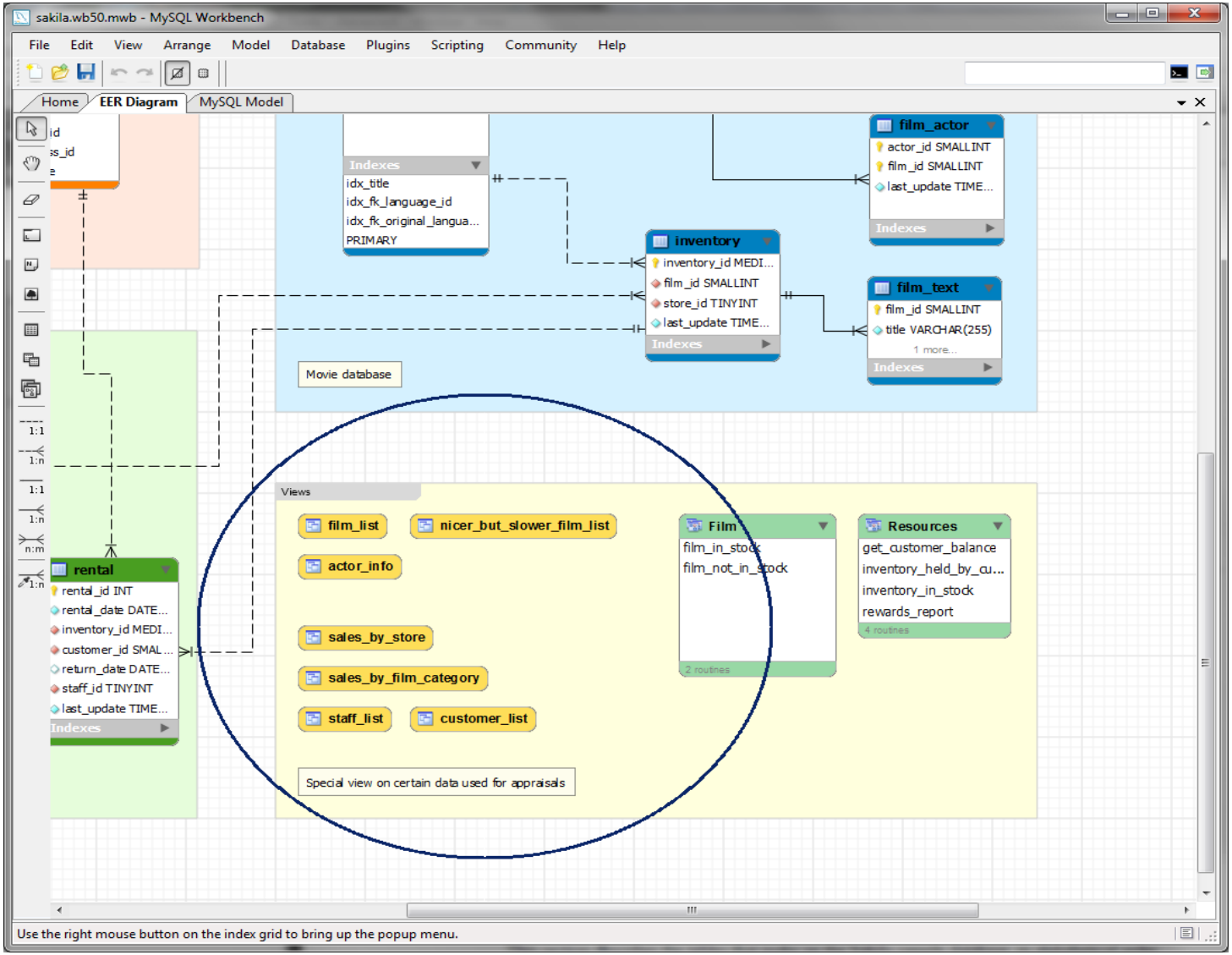
- rental
  - One row for each rental of each inventory item with information about who rented what item, when it was rented, and when it was returned
  - Referred to in the payment table
  - Refers to inventory, customer, and staff tables
- payment
  - One row for each payment made by a customer, with information such as amount, and the rental being paid for
  - Refers to customer, rental, and staff tables

Business-related Tables



Views

Views


sakila.wb50.mwb - MySQL Workbench
File Edit View Arrange Model Database Plugins Scripting Community Help
Home EER Diagram MySQL Model
Indexes
idx_title
idx_fk_language_id
idx_fk_original_langua...
PRIMARY
film_actor
actor_id SMALLINT
film_id SMALLINT
last_update TIME...
Indexes
inventory
inventory_id MEDI...
film_id SMALLINT
store_id TINYINT
last_update TIME...
Indexes
film_text
film_id SMALLINT
title VARCHAR(255)
1 more...
Indexes
Movie database
Views
film_list
nicer_but_slower_film_list
actor_info
sales_by_store
sales_by_film_category
staff_list
customer_list
Film
film_in_stock
film_not_in_stock
2 routines
Resources
get_customer_balance
inventory_held_by_cu...
inventory_in_stock
rewards_report
4 routines
Special view on certain data used for appraisals
rental
rental_id INT
rental_date DATE...
inventory_id MEDI...
customer_id SMAL...
return_date DATE...
staff_id TINYINT
last_update TIME...
Indexes
Use the right mouse button on the index grid to bring up the popup menu.

Views

- actor_info
- staff_list
- customer_list
- film_list
- nicer_but_slower_film_list
- sales_by_film_category
- sales_by_store

Stored Routines

Stored Procedures

- film_in_stock
  - Used to determine if there are any copies of a given film in stock at a given store
- film_not_in_stock
  - Used to determine if there are any copies of a given film not in stock at a given store
- rewards_report
  - Used to generate a customizable list of the top customers for the previous month

Stored Functions

- get_customer_balance
  - Returns the current amount owning on a specified customer's account
- inventory_held_by_customer
  - Returns the customer_id of the customer who has rented out the specified inventory items
- inventory_in_stock
  - Returns a boolean value indicating whether the inventory item specified in stock

Triggers

Triggers

- customer_create_date
  - Sets the create_date column of the customer table to the current time and date as rows are inserted
- payment_date
  - Sets the payment_date column of the payment table to the current time and date as rows are inserted
- rental_date
  - Sets the rental_date column of the rental table to the current time and date as rows are inserted

Triggers

- ins_film
  - Duplicates all INSERT operations on the film table to the film_text table
- upd_film
  - Duplicates all UPDATE operations on the film table to the film_text table
- del_film
  - Duplicates all DELETE operations on the film table to the film_text table

Examples

Example 1: Rent a DVD

- Assumption
  - 'inventory_id' of the DVD to be rented is known
  - 'customer_id' of the customer is known
  - 'staff_id' of the staff is known
- Tasks to be done
  - Confirm that the given inventory item is in stock, then insert a row into the rental table.
  - After the rental is created, insert a row into the payment table.
  - Depending on business rules, you may also need to check if the customer has an outstanding balance before processing the rental.

Example 1: Rent a Video

```
mysql> SELECT INVENTORY_IN_STOCK(10);
+-----------------------+
| INVENTORY_IN_STOCK(10) |
+-----------------------+
|                     1 |
+-----------------------+
1 row in set (0.14 sec)

mysql> INSERT INTO rental(rental_date, inventory_id, customer_id, staff_id)
    -> VALUES(NOW(), 10, 3, 1);
Query OK, 1 row affected (0.04 sec)

mysql> SELECT @balance := get_customer_balance(3, NOW());
+-----------------------------------------+
| @balance := get_customer_balance(3, NOW()) |
+-----------------------------------------+
|                                    4.99 |
+-----------------------------------------+
1 row in set (0.09 sec)

mysql> INSERT INTO payment (customer_id, staff_id, rental_id, amount, payment_date)
    -> VALUES(3,1,LAST_INSERT_ID(), @balance, NOW());
Query OK, 1 row affected (0.05 sec)
```

Example 2: Return a DVD

- Assumption
  - Each returned DVD should have 'inventory_id'
- Tasks to be performed -Update the rental table and set the return date. To do this, we first need to
  - Identify the rental_id to update based on the inventory_id of the item being returned
  - Depending on the situation we may then need to check the customer balance and perhaps process a payment for overdue fees by inserting a row into the payment table.

Example 2: Return a Video

```
mysql> SELECT rental_id
    -> FROM rental
    -> WHERE inventory_id = 10
    -> AND customer_id = 3
    -> AND return_date IS NULL;
+-----------+
| rental_id |
+-----------+
|     16050 |
+-----------+
1 row in set (0.00 sec)

mysql> UPDATE rental
    -> SET return_date = NOW()
    -> WHERE rental_id = @rentID;
Query OK, 0 rows affected (0.00 sec)

mysql> SELECT get_customer_balance(3, NOW());
+------------------------------+
| get_customer_balance(3, NOW()) |
+------------------------------+
|                         0.00 |
+------------------------------+
1 row in set (0.00 sec)
```

Thank you!

Sang Shin
http://www.javapassion.com
“Learning is fun!”