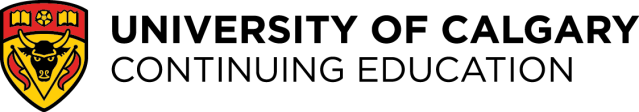
**Relational Databases Fundamentals**

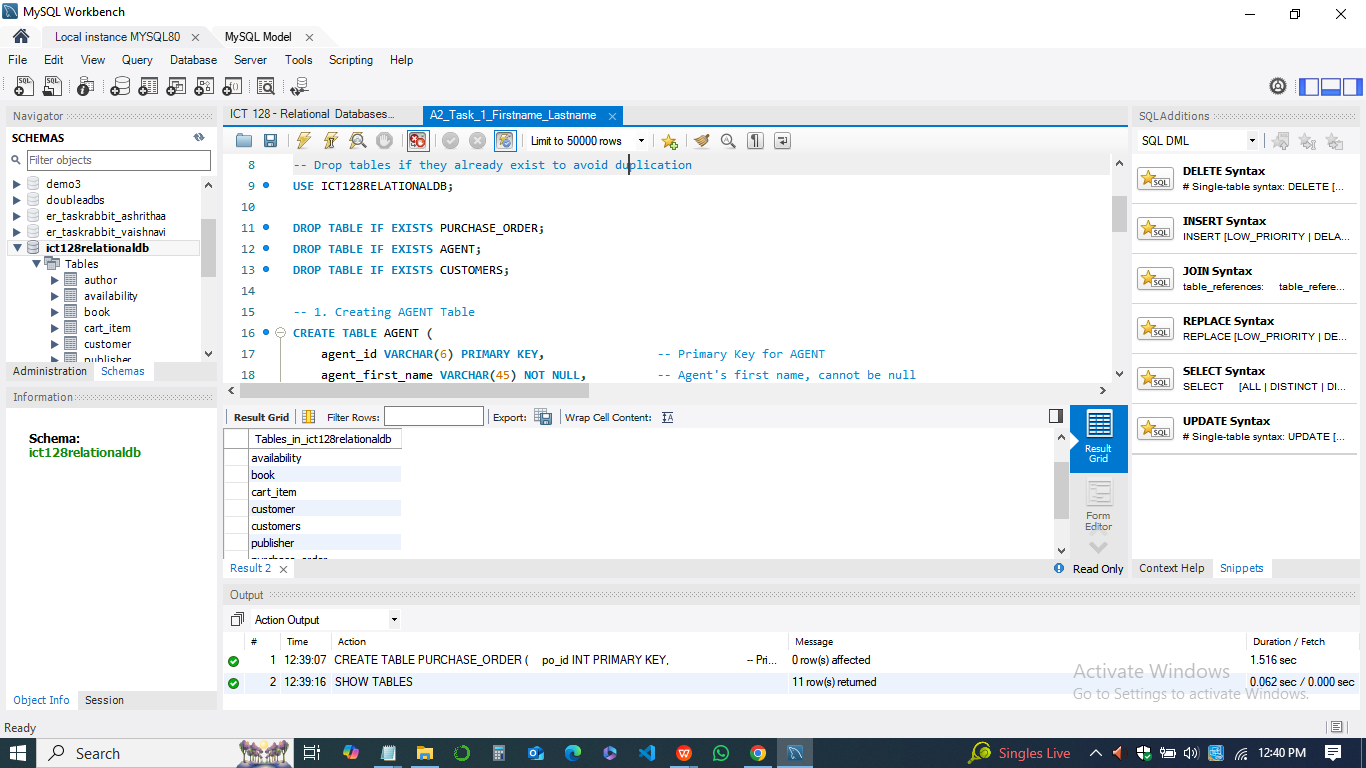


**Were Vincent**

**TASK 1**

**CREATING THE *PURCHASE\_ORDER, AGENT AND CUSTOMER TABLES***

In this assignment task 1, I created the necessary SQL Data Definition Language (DDL) statements to implement the relational database structure described in the Entity-Relationship Diagram (ERD). The project involved defining tables for "AGENT," "CUSTOMER," and "PURCHASE\_ORDER," each with specific primary keys, foreign keys, attributes, and constraints to ensure data integrity.



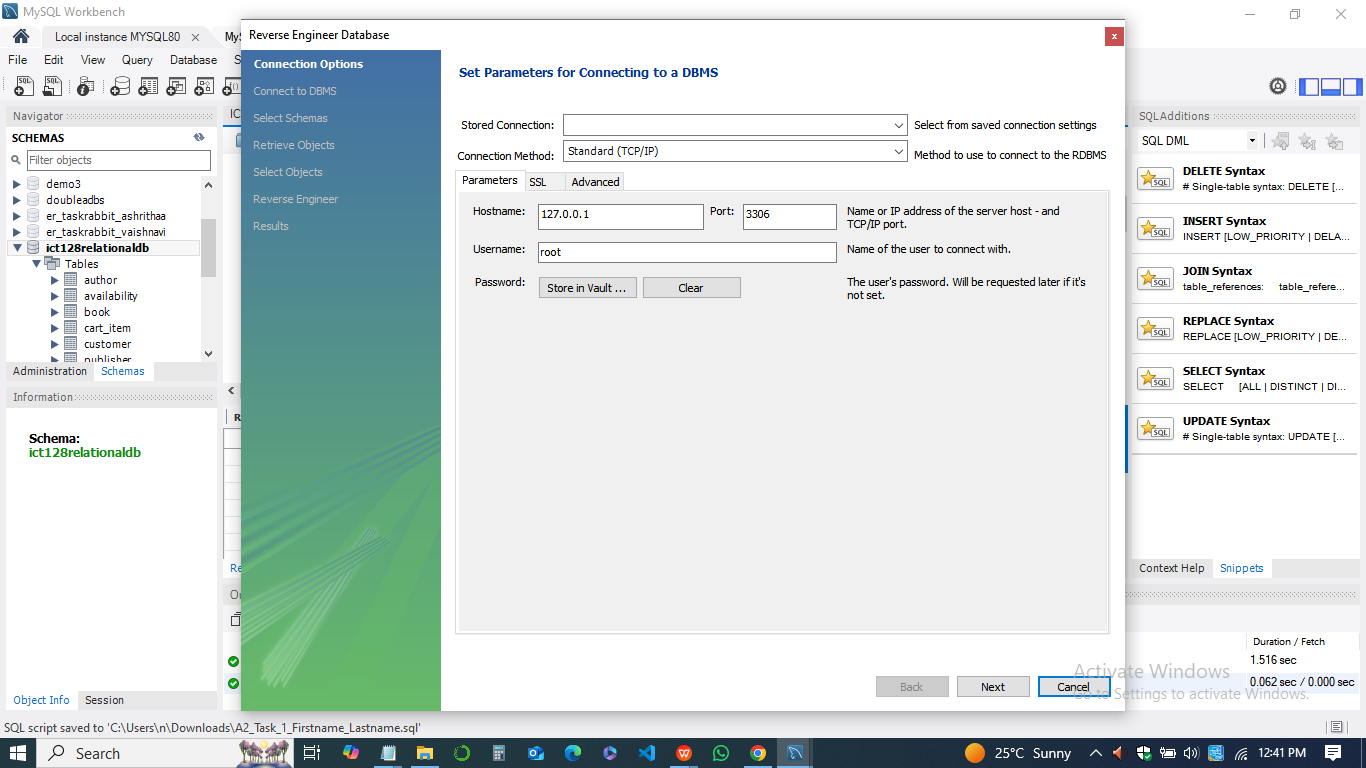
First, I began the script by dropping any pre-existing tables, AGENT, CUSTOMER, and PURCHASE\_ORDER, to prevent errors when the script is rerun. This step helps ensure that the database remains clean, allowing us to build the tables from scratch.

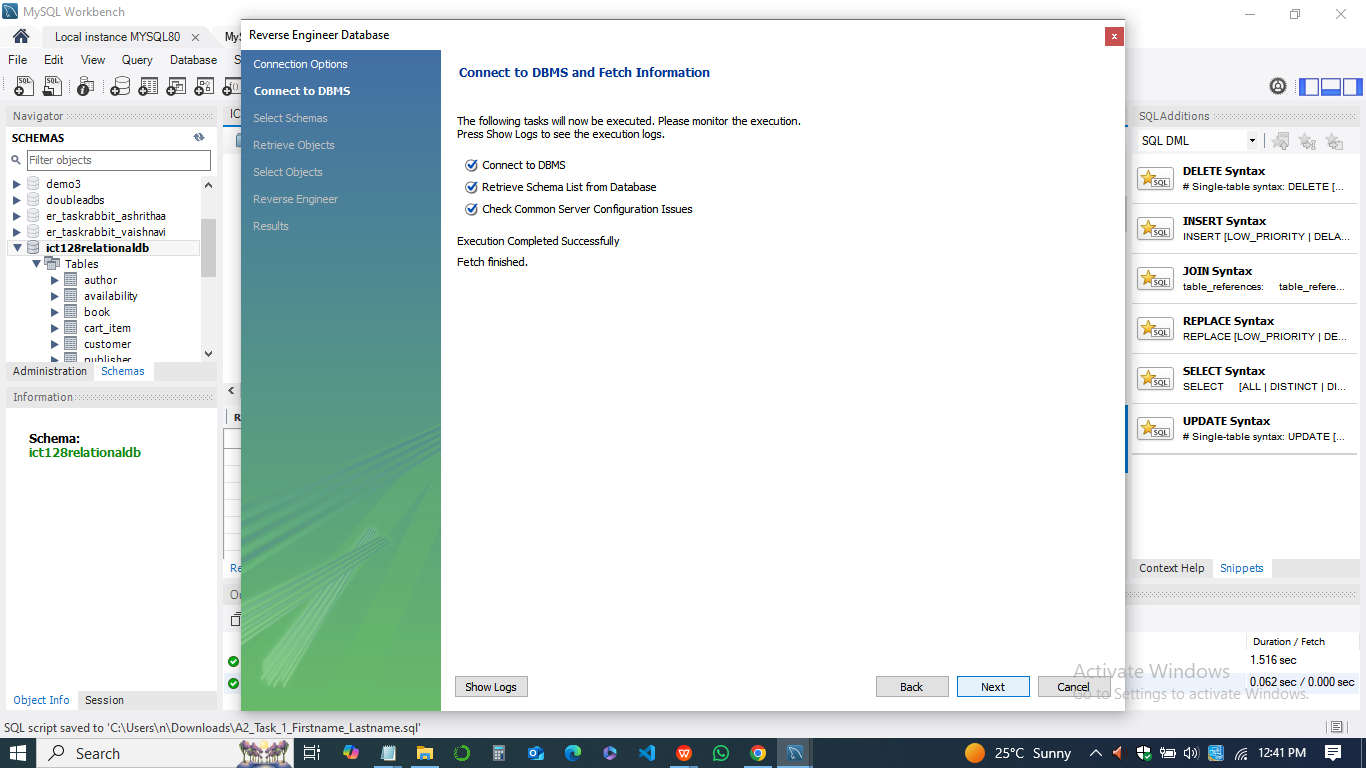
The AGENT table was then created to store sales agent information, with essential attributes such as agent\_id (primary key), first name, last name, area, commission rate, phone number, and country code. Each attribute was given an appropriate data type, and several fields were marked as "NOT NULL" to ensure they are required when inserting data.

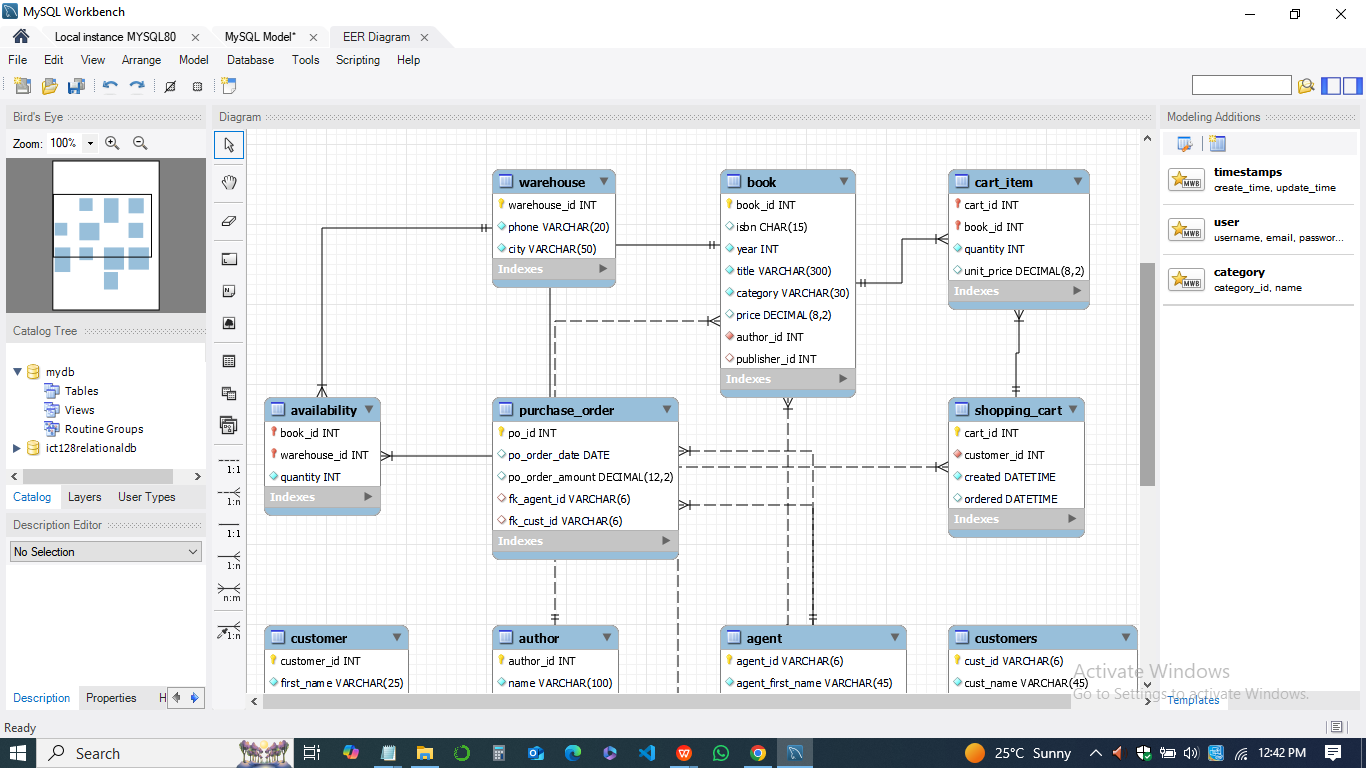
Next, I defined the CUSTOMER table to capture customer details, setting cust\_id as the primary key and including fields for customer name, city, area, country, phone number, and opening account amount. Like the agent table, many of these fields were set to "NOT NULL" to enforce data completeness.

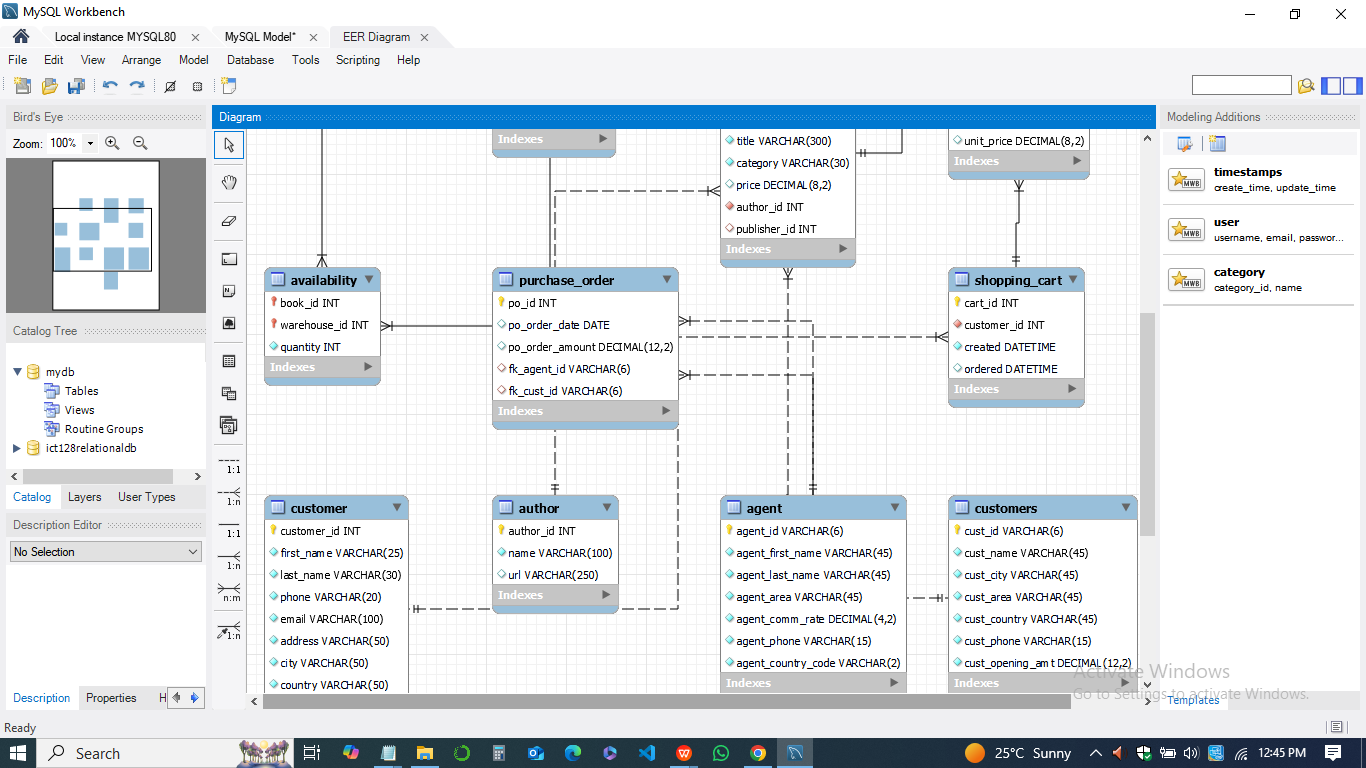
Finally, I created the PURCHASE\_ORDER table, which records purchase orders associated with specific agents and customers. The po\_id serves as the primary key, and foreign keys fk\_agent\_id and fk\_cust\_id reference the AGENT and CUSTOMER tables, respectively. These foreign keys establish one-to-many relationships between agents, customers, and their purchase orders, with ON DELETE SET NULL constraints to maintain referential integrity.

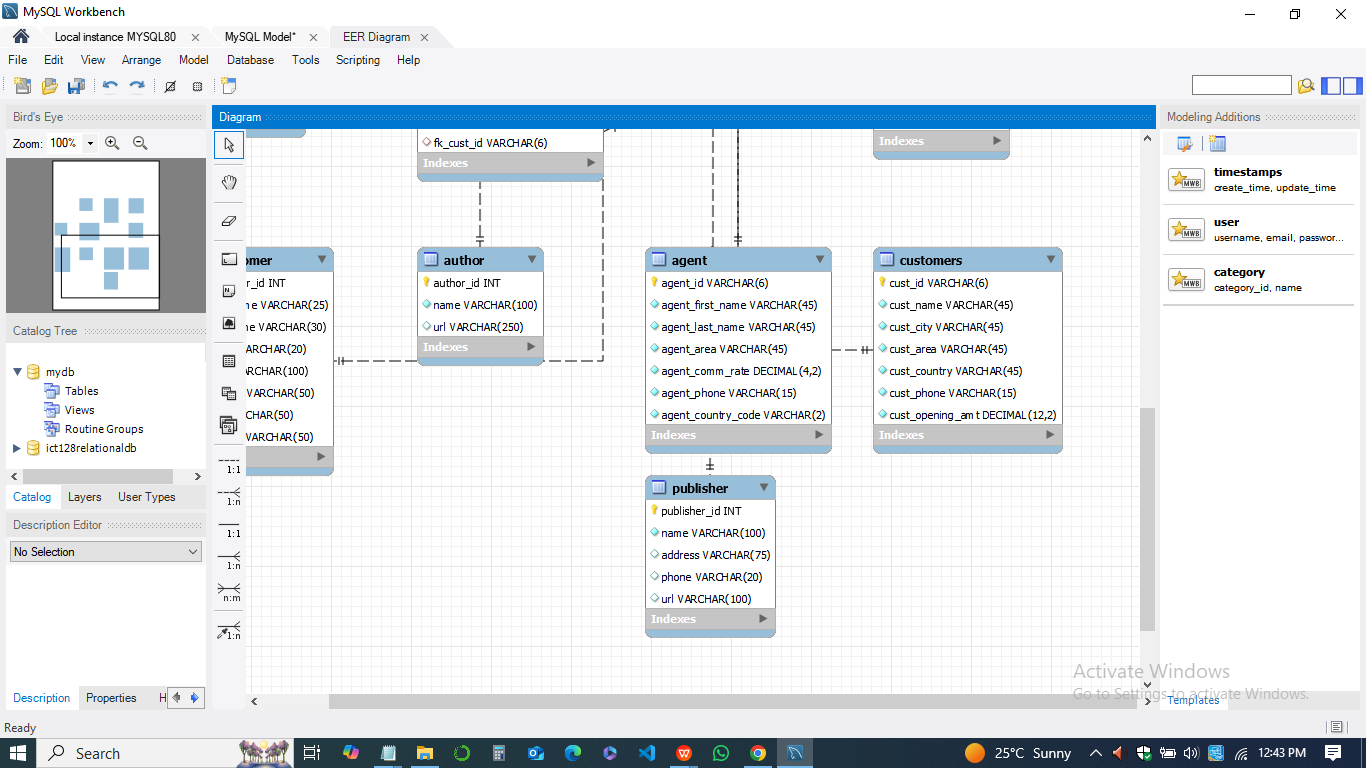
**BY REVERSE ENGINEER**











This approach ensures that the database structure supports the intended relationships between entities, enabling robust data storage and retrieval capabilities in MySQL Workbench. The completed script is saved as *A2\_Task\_1\_Firstname\_Lastname.sql* for submission.