**Upload Assignment: [Group] Database Security and Compliance Implementation**

**ITMS 528-01 – Database Security**

**Professor name – Dr. Maurice Dawson**

**Group Members:**

**Onkar Rahane – A20541728**

**Hrishikesh Pawar – A20543095**

**Prashant Mane – A20536534**

**Task 1: Database Creation and Security Controls**

**Database Design and Implementation.**

Successfully created ecommerce database with tables such as users, products, orders, order\_items, categories, reviews and addresses. The table structure was as follows:

User Table:

* user\_id (Primary Key)
* username
* email
* password

Products Table:

* product\_id (Primary Key)
* name
* description
* price
* category

Order Table:

* order\_id (Primary Key)
* user\_id (Foreign Key referencing User table)
* order\_date
* total\_amount

Order\_Items Table:

* orderitem\_id (Primary Key)
* order\_id (Foreign Key referencing Order table)
* product\_id (Foreign Key referencing Products table)
* quantity
* price

Categories Table:

* category\_id (Primary Key)
* category\_name

Addresses Table:

* address\_id (Primary Key)
* user\_id (Foreign Key referencing User table)
* street
* city
* state
* zip\_code

Reviews Table:

* review\_id (Primary Key)
* user\_id (Foreign Key referencing User table)
* product\_id (Foreign Key referencing Products table)
* rating
* comment

With this structure, we can perform various queries and operations, such as retrieving a user's order history, getting all products in a specific category, calculating the total revenue, and more. Python script was used to interact with the MySQL server, allowing us to perform CRUD (Create, Read, Update, Delete) operations on the database.

Python Script to interact with the database server through linux:

A screenshot of a computer screen

Description automatically generated A computer screen with text on it

Description automatically generated

The above code shows the table structure to be inserted into the database through mysql server into ecommerce database.

A screenshot of a computer

Description automatically generated A screen shot of a computer

Description automatically generated

By using faker library we used it to populate the database. All the 7 tables were populated with genuine examples and information. Total of 1100 entries were filled.

A computer screen with text on it

Description automatically generated by using mysql.connector library we successfully entered all the information into the database schema of ecommerce.

**Implementation of Security Controls:**

* AC-2 ACCOUNT MANAGEMENT

The implementation of NIST SP 800-53 AC-2 within the e-commerce database, particularly in the context of creating distinct user roles such as company\_head, project\_manager, and system\_admin, involves several key principles to ensure effective account management and enhance overall security.

Role-Based Access Control (RBAC):

Objective: Assign users to specific roles with corresponding privileges based on their job responsibilities.

Implementation: Define clear roles such as company\_head, project\_manager, and system\_admin, each with a unique set of permissions. Establish RBAC policies that restrict access to database resources according to these roles. This ensures that users have the necessary privileges to perform their tasks without unnecessary access to sensitive information.

Password Policies:

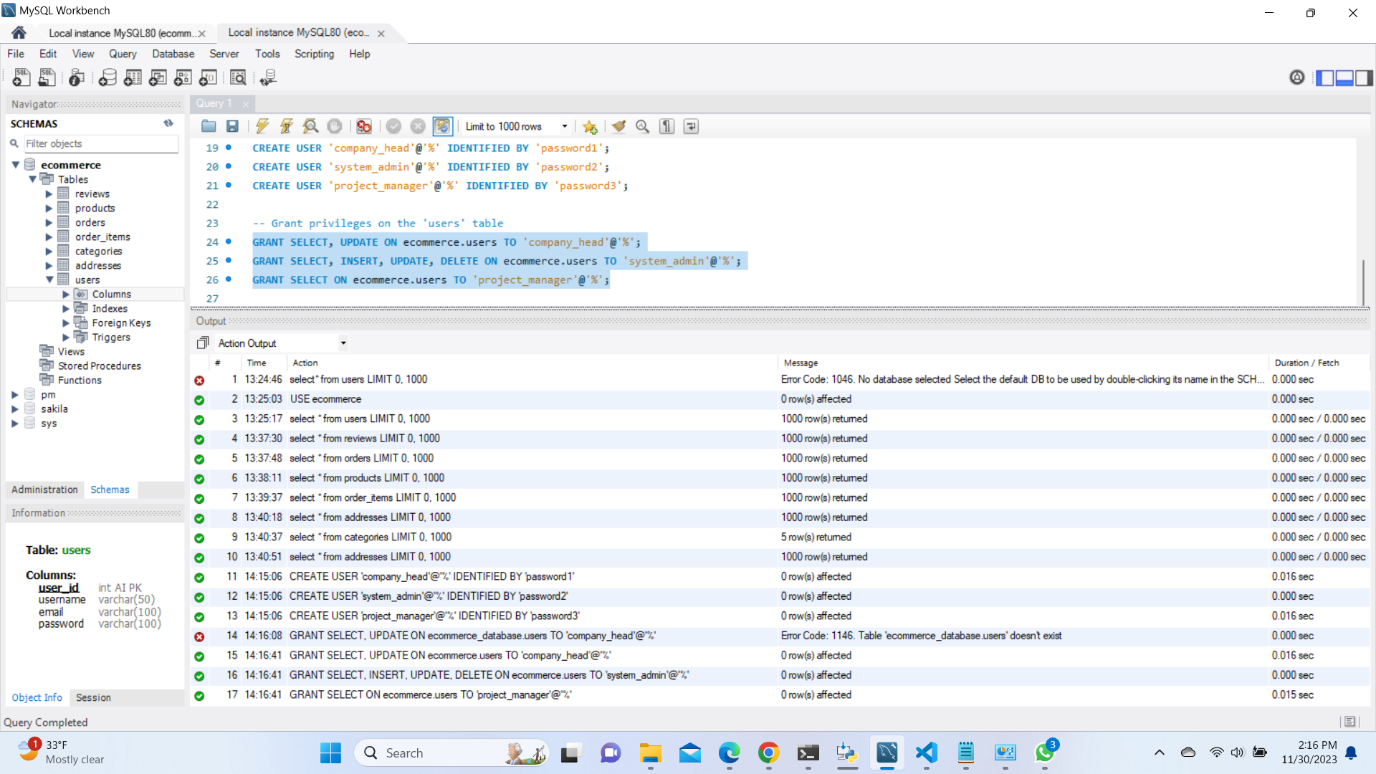
Objective: Enforce strong password policies to enhance the security of user accounts.

Implementation: Implement password complexity requirements for all user roles, including company\_head, project\_manager, and system\_admin. This involves specifying minimum length, character types, and regular password updates. By adhering to robust password policies, the system mitigates the risk of unauthorized access due to weak or compromised passwords.

A screenshot of a computer

Description automatically generated

The above image shows the implementation of AC-2 for creating users in the ecommerce database.



The GRANT query was used to assign them privileges in the database of ecommerce. According to these roles they will be able to execute their duties.

* AC-5 SEPARATION OF DUTIES

The implementation of NIST SP 800-53 AC-5 within the e-commerce database, focusing on the concept of Separation of Duties (SoD), involves strategically assigning different privileges to existing user roles (company\_head, project\_manager, and system\_admin) to prevent conflicts of interest and reduce the risk of unauthorized activities. Here's a short theory about the implementation:

Role Definition and Responsibilities:

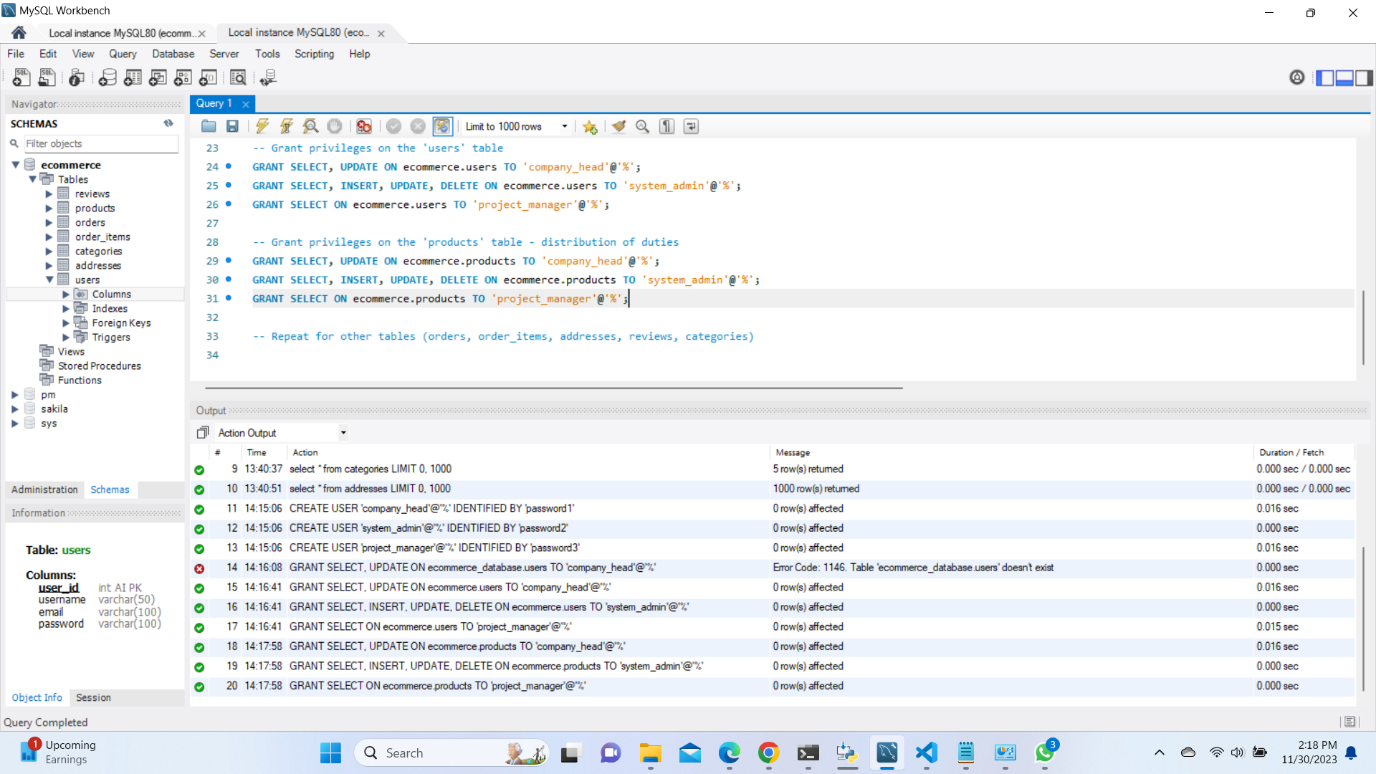
Objective: Clearly define the responsibilities of each user role to ensure a clear understanding of their duties.

Implementation: For each user role (company\_head, project\_manager, and system\_admin), meticulously outline the specific duties they are entrusted with. This includes tasks related to account management, product management, order processing, system configuration, and any other relevant activities within the e-commerce platform.

Privilege Assignment Based on Responsibilities:

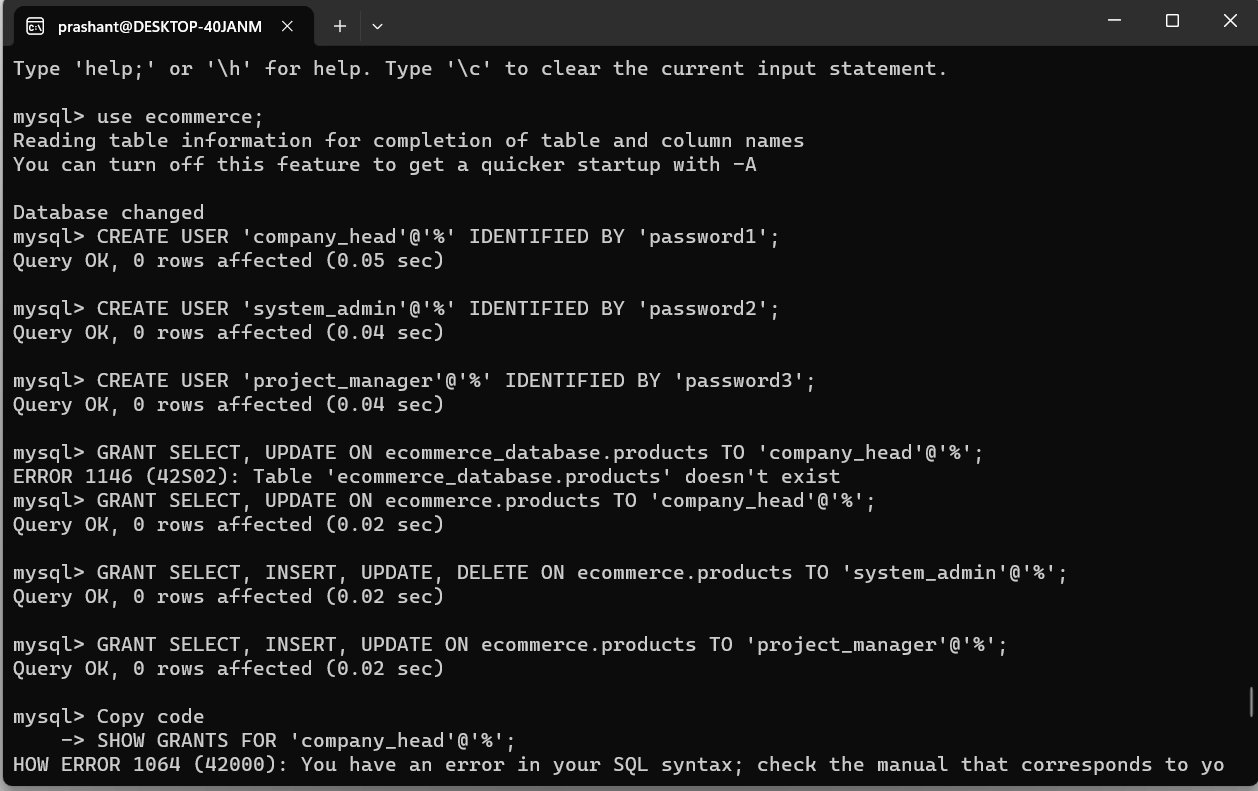
Objective: Align user privileges with their designated responsibilities, adhering to the principle of least privilege.

Implementation: Assign privileges to each user role based on their defined responsibilities. Company\_head may have high-level access, overseeing all aspects of the e-commerce platform, while project\_manager focuses on product and order management. System\_admin, on the other hand, may have exclusive access to system configuration settings. By aligning privileges with responsibilities, the system ensures that users can perform their duties without unnecessary access to unrelated functions.

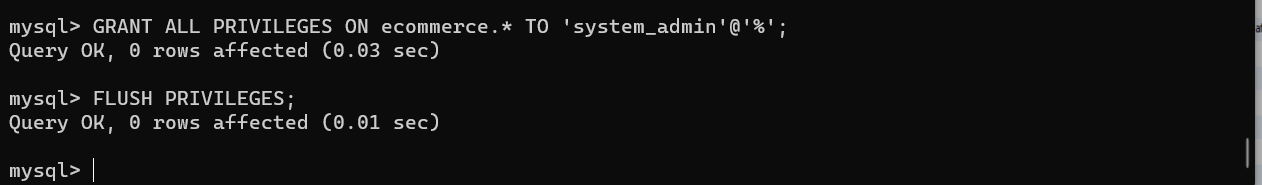


According to the above image we assigned different privileges to the table ‘products and users’. This is how the separation of duties were preformed in the database of ecommerce.

This implementation is also done through mysql server in linux using ubuntu as shown below.



Flushed the privileges:



* AC-8 SYSTEM USE NOTIFICATION

The implementation of NIST SP 800-53 AC-8, focusing on System Use Notification, involves the integration of a notification banner displayed before user login to the database. This serves as an important mechanism to inform users about their responsibilities and the acceptable use of the system. Here's a short theory about the implementation:

Notification Banner Content:

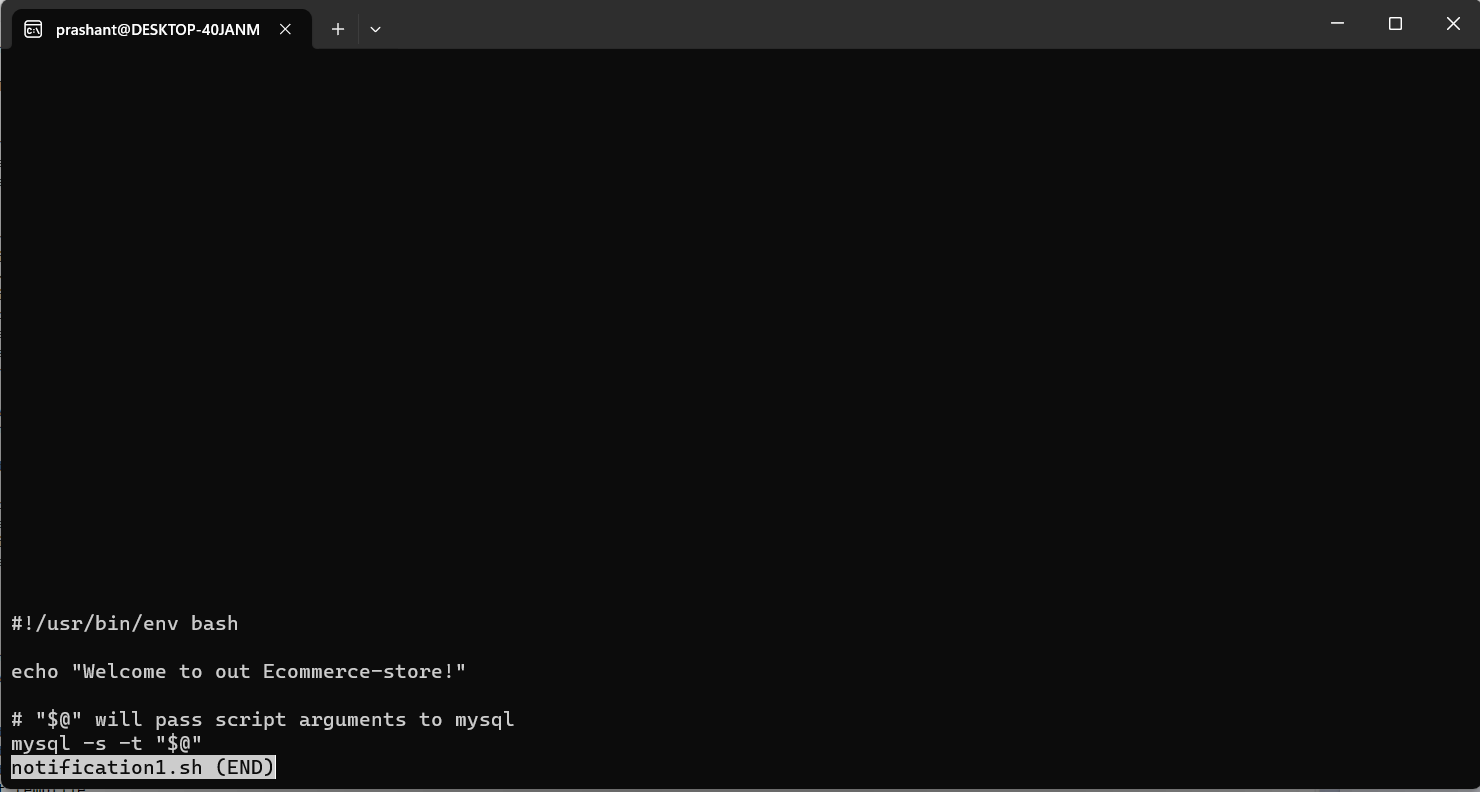
Objective: Inform users of their responsibilities and the acceptable use of the e-commerce database.

Implementation: Develop a clear and concise notification message to be displayed in the banner. This message should include information on authorized access, the monitoring of activities, and any relevant legal or policy considerations. The content should be crafted to effectively communicate the importance of adhering to security and usage policies.

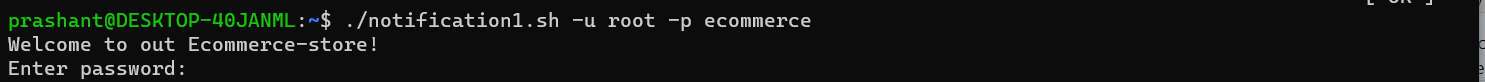
Banner Placement:

Objective: Ensure the notification banner is prominently displayed before user login.

Implementation: Integrate the notification banner into the login interface, making it the first element users encounter when accessing the system. This ensures that users are immediately informed of the system use policies and responsibilities before proceeding with the login process.



This is how the script was written. The objective was to display a notification banner as “Welcome to our Ecommerce-store!” after every login attempt.



This is how the notification was displayed.

* AC-7 UNSUCCESSFUL LOGON ATTEMPTS

The implementation of NIST SP 800-53 AC-7, focusing on Unsuccessful Logon Attempts, involves deploying measures to detect and respond to repeated unsuccessful login attempts, enhancing the security of the e-commerce database. In this scenario, we'll illustrate the implementation for the user role project\_manager and demonstrate a timeout mechanism after three unsuccessful login attempts:

Detection of Unsuccessful Logon Attempts:

Objective: Identify and track unsuccessful login attempts for the project\_manager user role.

Implementation: Integrate a mechanism within the authentication process to monitor and count unsuccessful login attempts specifically for the project\_manager role. Utilize log files or an intrusion detection system to record these events, ensuring that the system can identify patterns indicative of potential unauthorized access.

Threshold Configuration:

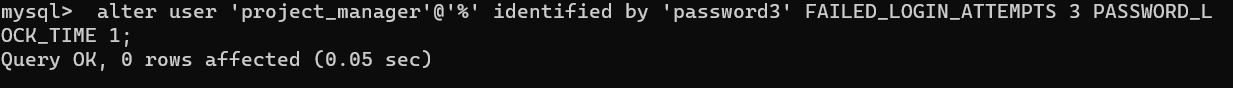
Objective: Set a threshold for the number of allowable unsuccessful login attempts.

Implementation: Establish a threshold, such as three unsuccessful login attempts, beyond which the system takes preventive action. Configuring this threshold is a crucial step to strike a balance between security and user convenience.

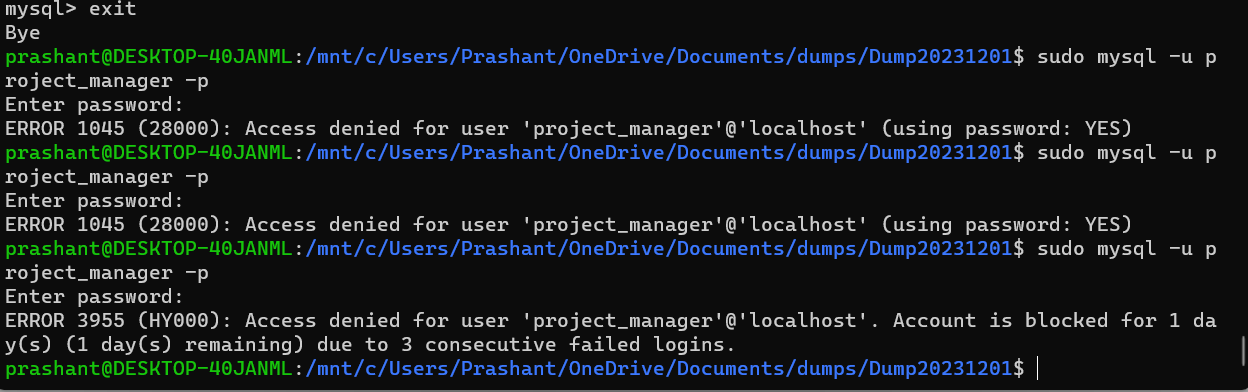
Timeout Mechanism:

Objective: Implement a timeout mechanism after reaching the threshold to prevent brute-force attacks.

Implementation: Upon the third unsuccessful login attempt for the project\_manager role, enforce a temporary account lockout or a timeout period. This prevents further login attempts for a predefined duration, thwarting potential brute-force attacks and enhancing overall security.



Implemented this timeout query in the mysql server on the ecommerce database for 3 failed login attempts and password lock.



The above screen shot shows the demonstration of failed login attempts on user – project\_manager.

* AT-4 SECURITY TRAINING RECORDS

The implementation of NIST SP 800-53 AT-4, which focuses on Security Training Records, is a critical aspect of maintaining a secure e-commerce database system. This control ensures that individuals with access to the system receive appropriate security training, and their training records are effectively managed. Here's a short theory about the implementation:

Training Program Design:

Objective: Develop a comprehensive security training program tailored to the roles and responsibilities of individuals accessing the e-commerce database.

Implementation: Create a structured training program covering essential security topics such as data protection, access controls, password management, and incident response. Tailor the content to specific roles within the organization, ensuring that individuals receive training relevant to their responsibilities.

User Enrollment and Tracking:

Objective: Enroll users in the security training program and establish a tracking mechanism for their progress.

Implementation: Integrate a user enrollment system that includes all individuals requiring access to the e-commerce database. Develop a tracking mechanism to monitor and record each user's progress through the training program, ensuring that all relevant training modules are completed.

Record Keeping:

Objective: Maintain accurate and up-to-date records of individual security training achievements.

Implementation: Establish a database system to store training records, including details such as user names, completion dates, and specific training modules completed. Regularly update these records to reflect the most recent training achievements and ensure accuracy during audits or compliance checks.

* AU-8 TIME STAMPS

The implementation of NIST SP 800-53 AU-8, focusing on Time Stamps, is crucial for ensuring accurate and reliable tracking of events within the e-commerce database, particularly in the context of the Orders table. By utilizing date-time data types for the order\_date column, this implementation enhances the system's ability to generate meaningful audit trails and supports forensic analysis. Here's a short theory about the implementation:

Timestamp Integration:

Objective: Integrate timestamping into the Orders table to record the date and time of significant events, particularly order creation and modification.

Implementation: Configure the order\_date column in the Orders table to utilize a date-time data type. This allows the database to automatically capture and store the precise timestamp whenever an order is created or modified. The inclusion of timestamps ensures a granular record of events associated with orders, facilitating auditability and forensic analysis.

Consistent Timestamp Format:

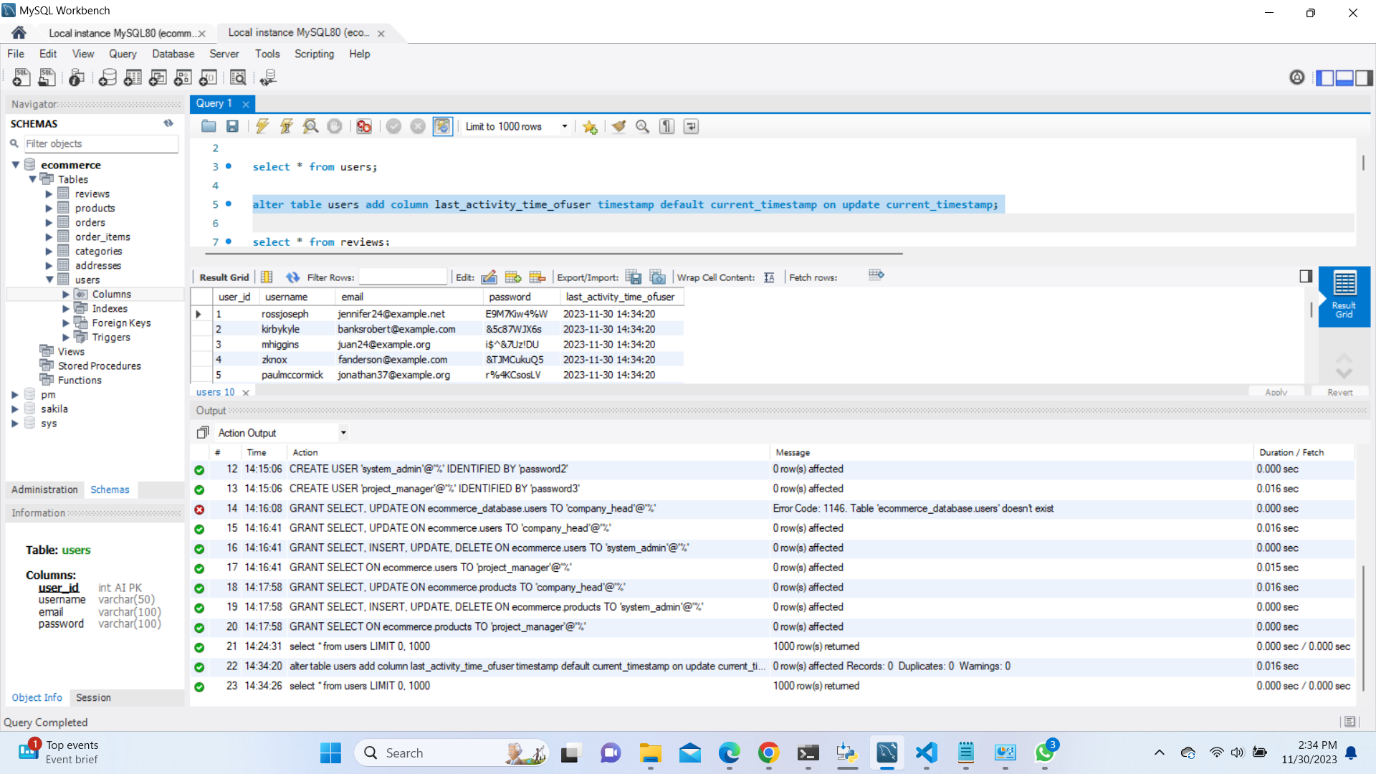
Objective: Maintain a standardized timestamp format across the Orders table for consistency and ease of interpretation.

Implementation: Enforce a consistent timestamp format, such as ISO 8601 (YYYY-MM-DDTHH:MM:SS), throughout the Orders table. This standardization facilitates ease of interpretation, ensuring that timestamps are uniformly presented for all orders and simplifying the process of querying and analyzing the data.

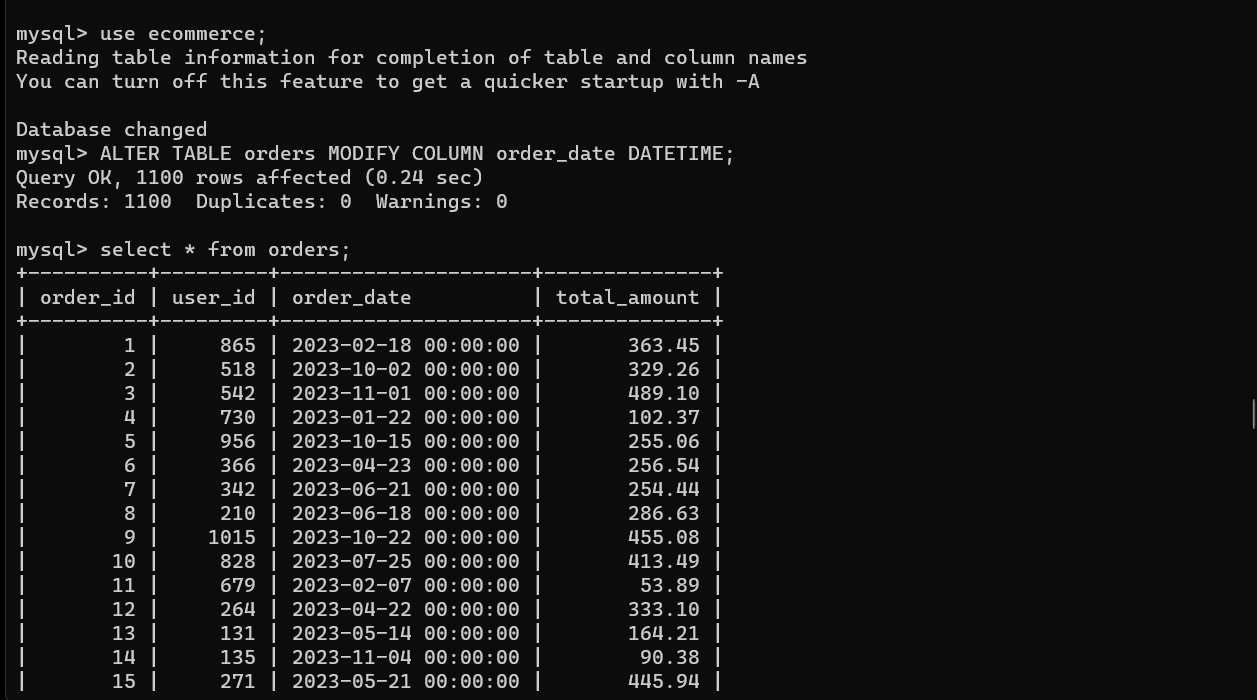
Automated Timestamp Generation:

Objective: Automate the generation of timestamps to eliminate reliance on manual input and reduce the risk of errors.

Implementation: Configure the database system to automatically generate timestamps for the order\_date column upon order creation or modification. This automated approach minimizes the possibility of human error, ensures accuracy, and streamlines the timestamping process.



Implementation of AU-8 on users table to check for their last activity.



implementation of of AU-8 on orders table to check for order date.

* CA-5 PLAN OF ACTION AND MILESTONES

The implementation of NIST SP 800-53 CA-5, focusing on the Plan of Action and Milestones (POA&M), is a critical component of managing and addressing security weaknesses within the e-commerce database system. The POA&M serves as a roadmap for identifying, prioritizing, and mitigating security vulnerabilities. Here's a short theory about the implementation:

Vulnerability Assessment and Identification:

Objective: Conduct regular vulnerability assessments to identify security weaknesses within the e-commerce database.

Implementation: Implement a systematic process for regularly assessing the security posture of the database system. Utilize tools and methodologies to identify vulnerabilities, including those related to data integrity, access controls, and configuration settings.

Prioritization of Security Weaknesses:

Objective: Prioritize identified security weaknesses based on the level of risk and potential impact.

Implementation: Develop a risk-based approach to prioritize security weaknesses. Consider factors such as the severity of the vulnerability, the likelihood of exploitation, and the potential impact on the confidentiality, integrity, and availability of the e-commerce database.

Documentation of Security Weaknesses:

Objective: Document identified security weaknesses in a centralized repository.

Implementation: Establish a dedicated repository or database to document and track security weaknesses. Each entry in the repository should include details such as the nature of the vulnerability, its risk level, and any relevant contextual information. This documentation forms the foundation of the POA&M.

Development of Mitigation Strategies:

Objective: Formulate mitigation strategies to address identified security weaknesses.

Implementation: For each documented security weakness, develop specific and actionable mitigation strategies. These strategies should outline steps to remediate or mitigate the vulnerabilities effectively. Consider involving relevant stakeholders, including IT personnel, database administrators, and security professionals, in crafting these strategies.

Assignment of Responsibility:

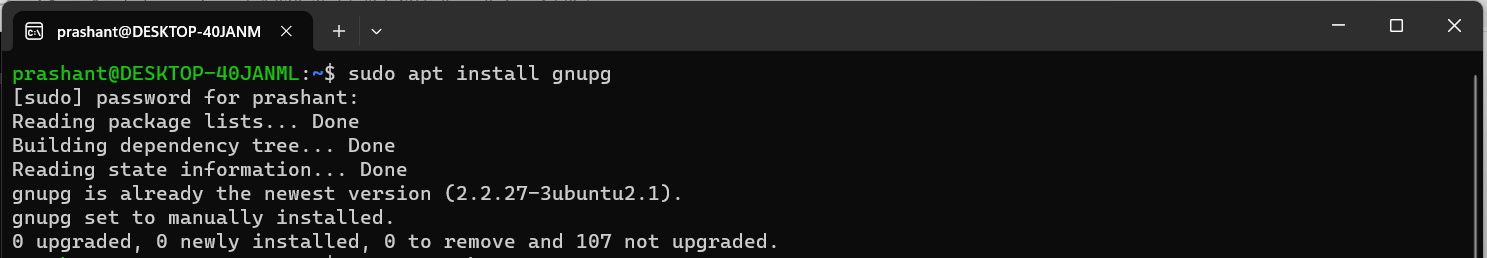
Objective: Clearly assign responsibility for implementing mitigation strategies.

Implementation: Define and assign responsibility for each mitigation strategy to the appropriate individuals or teams. Establish clear lines of communication and accountability to ensure that the necessary actions are taken promptly and efficiently.

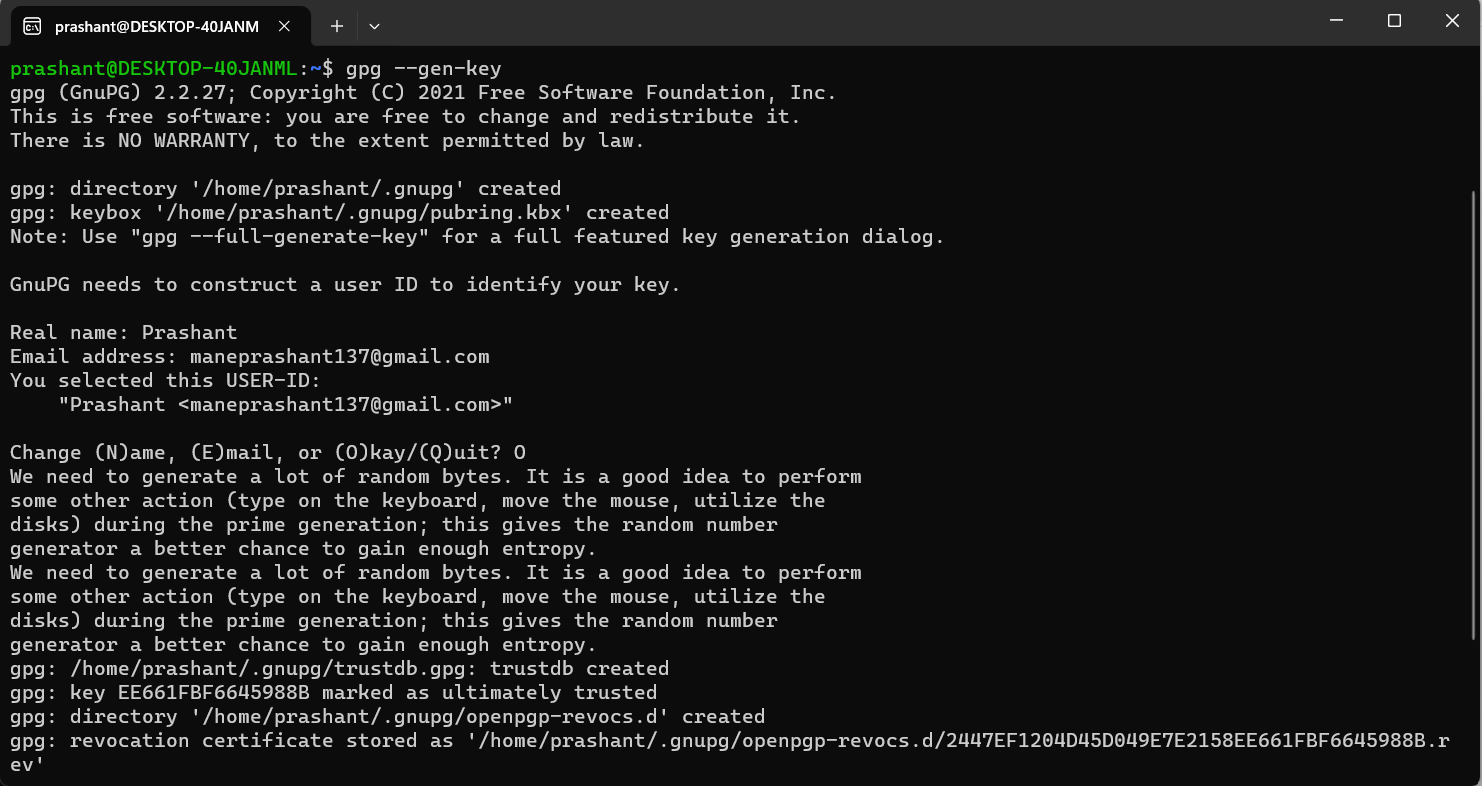
**Task 2: Data Encryption**

**Data Encryption Implementation**

**For data at rest -**

****

Install gnupg in your linux terminal. The command for the same is shown above.



While generating gnupg key, make sure to note the passphrase which will be entered. This passphrase is important while decrypting which is demonstrated further.

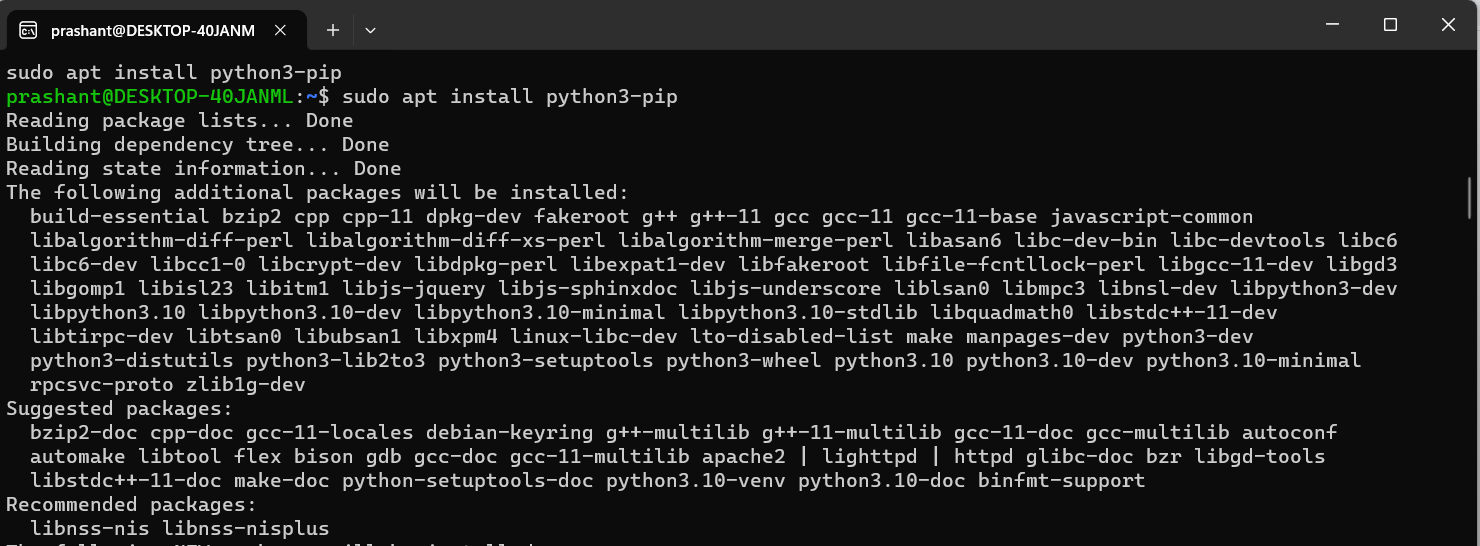


Now, use –list-keys to view the keys.

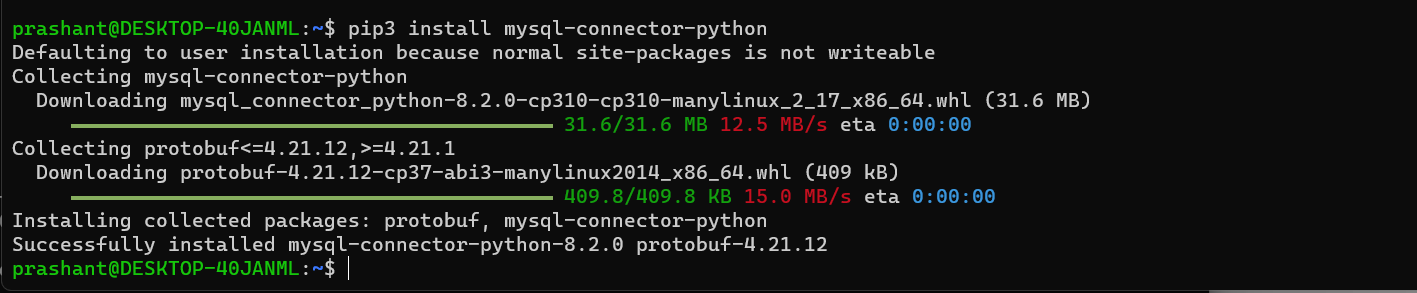
A computer screen with white text

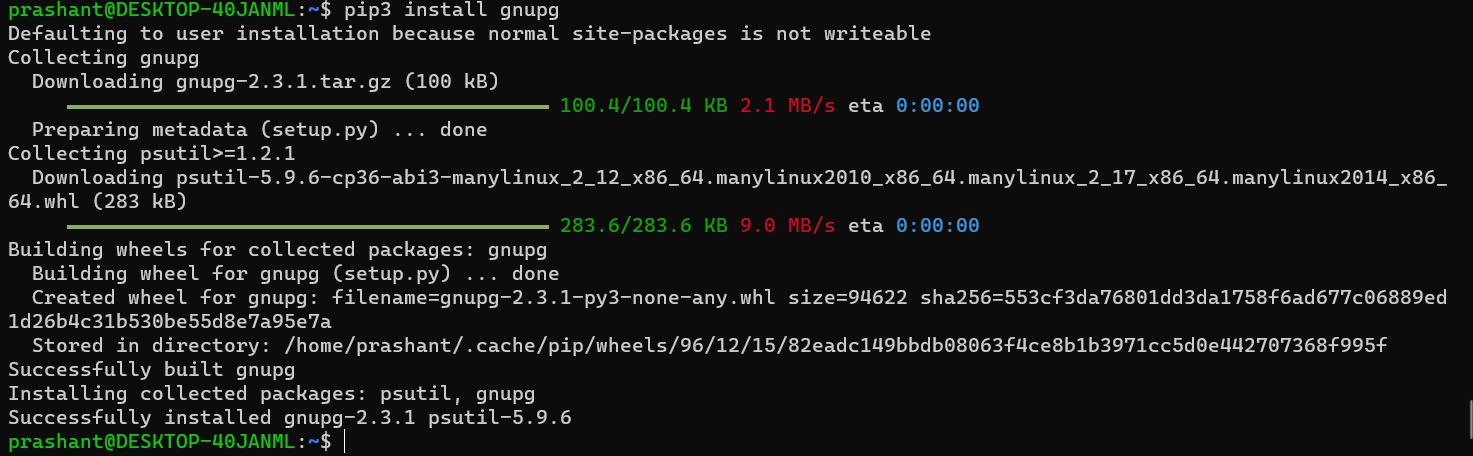
Description automatically generated

Install python libraries which will be required to run the python script to apply encryption on specific fields. Command – “pip3 install gnupg” and “pip3 install mysql-connector-python”



Install mysql-connector-python





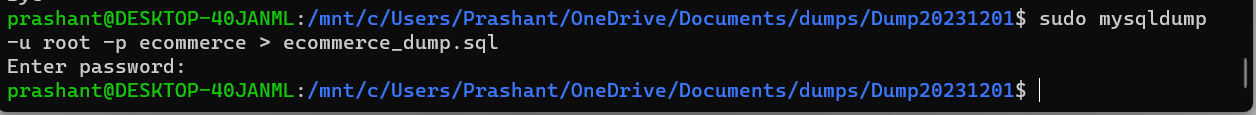
Install gnupg as shown above.

Now, Export the generated key into your local system using the email id provided while generating the key through gnupg. The command for that is shown below:

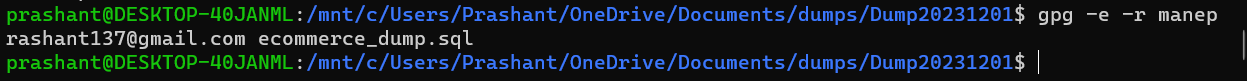


Now, encrypt the entire database file using GnuPG

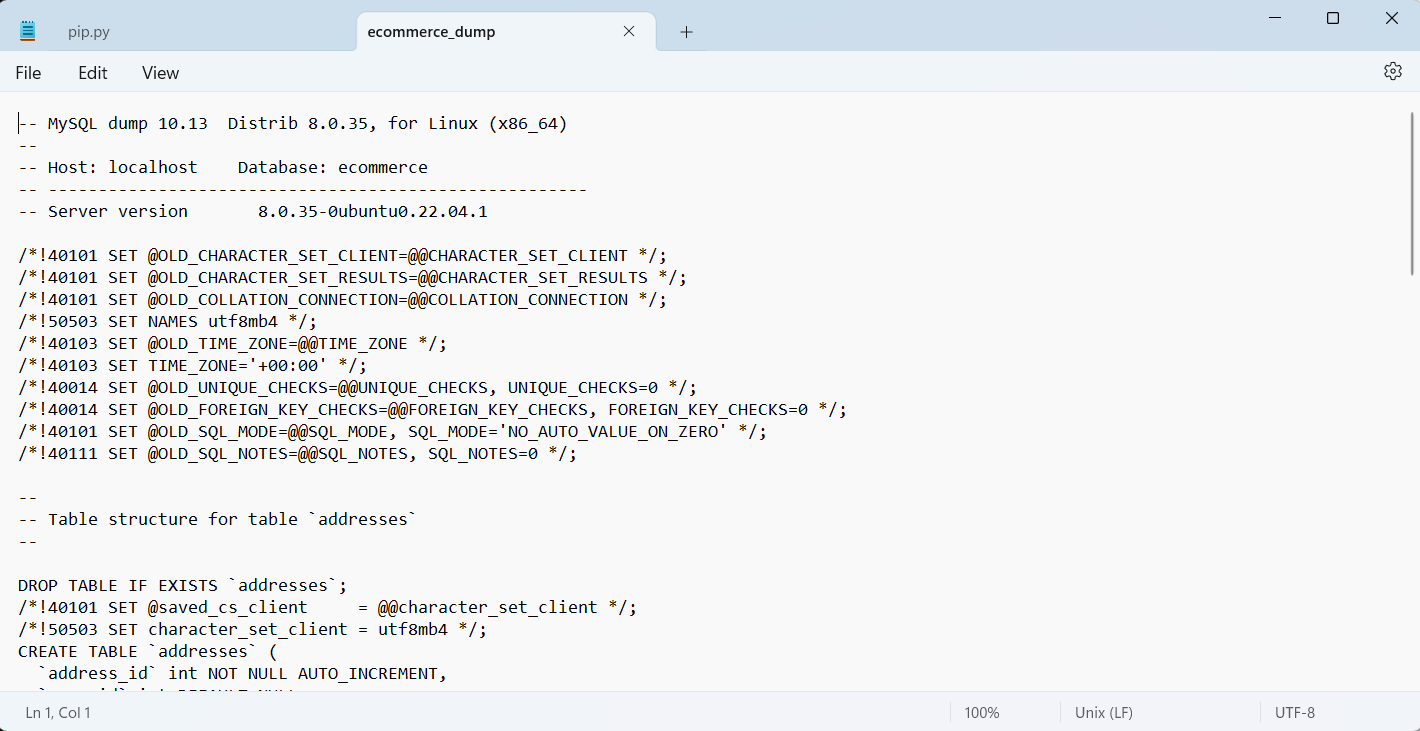
But first export the database into .sql file format to perform the encryption using gnuPG



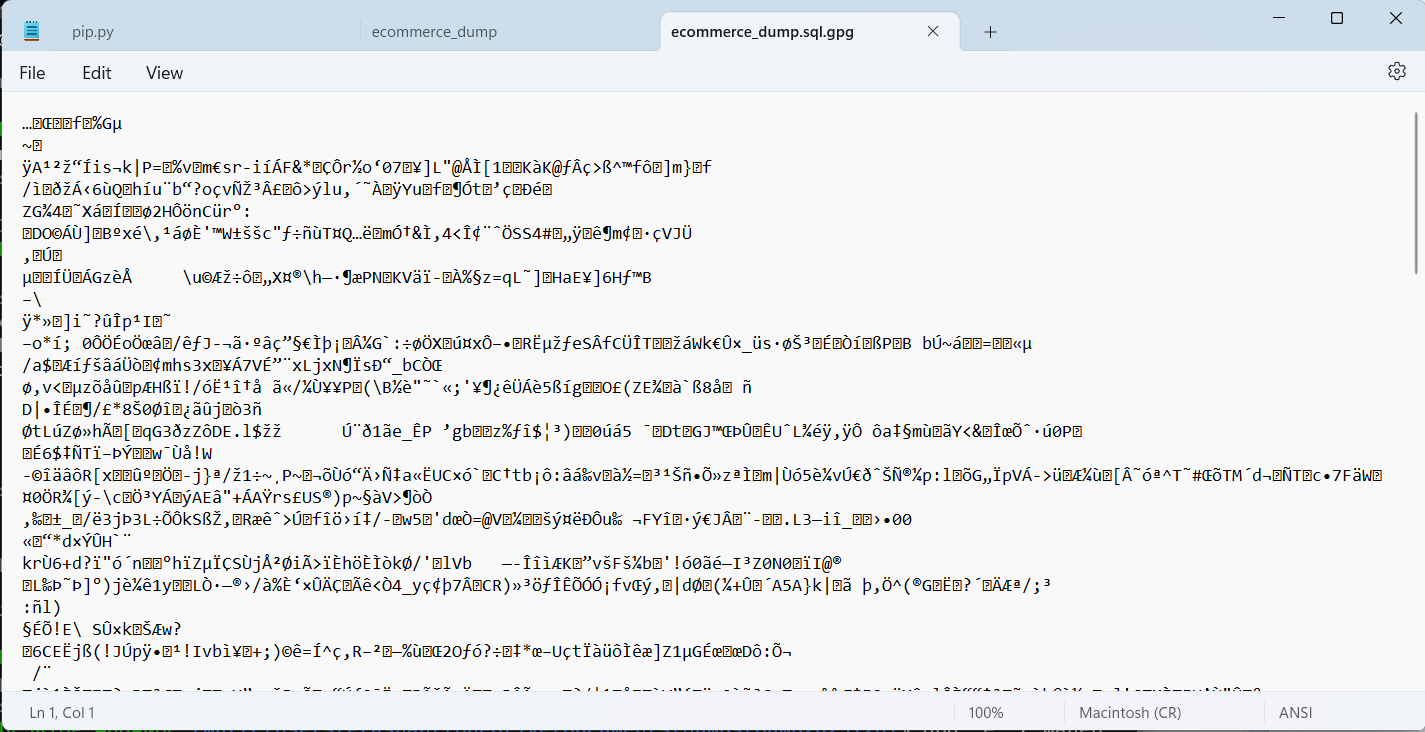
For encryption use -



Now, after the encryption is performed correctly view the .sql file.



This was before encryption.

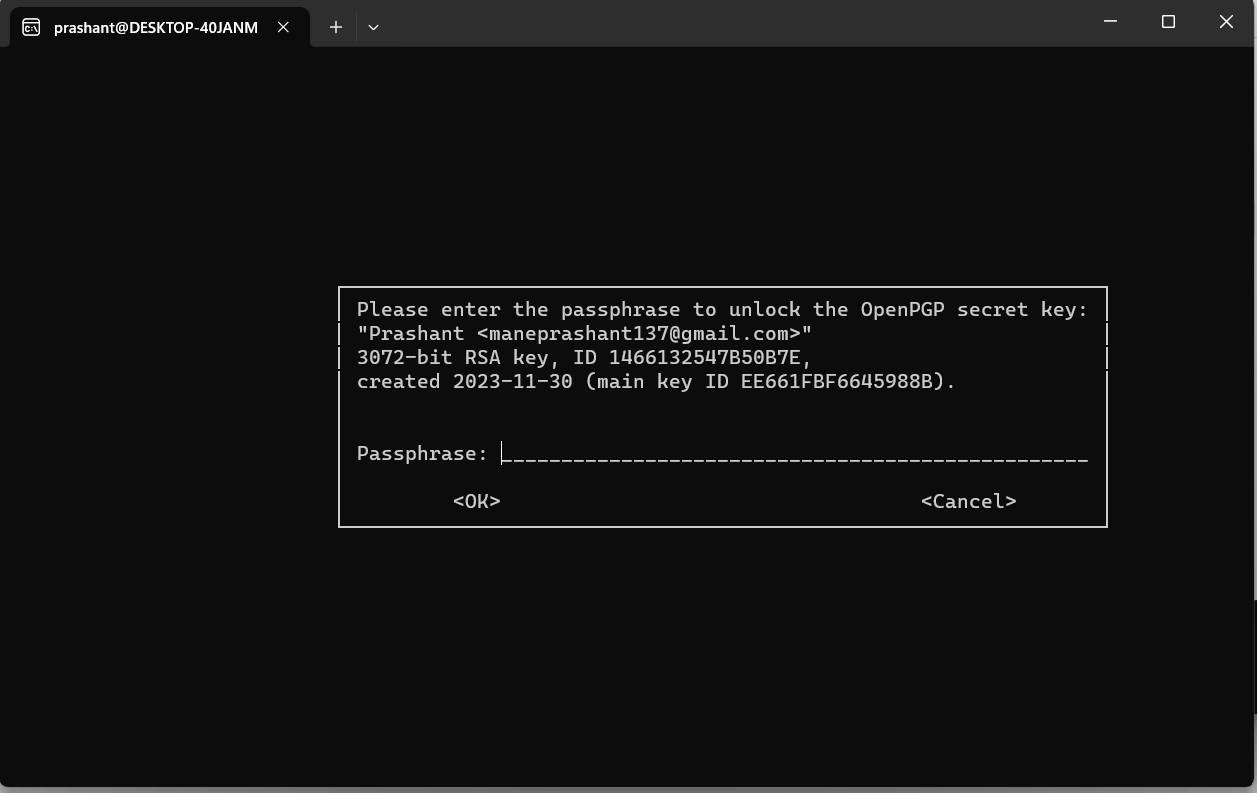


This is the same file after applying gnupg encryption.

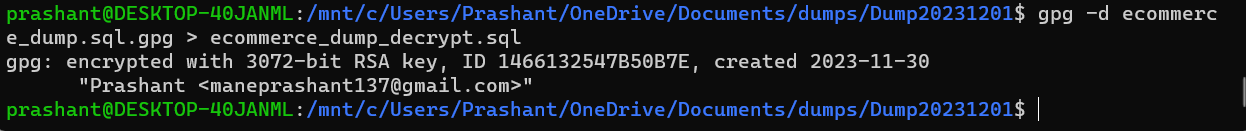
Now, we’ll see the procedure for decryption:

This is shown by the following command. The command uses the created gpg file while using for decryption.

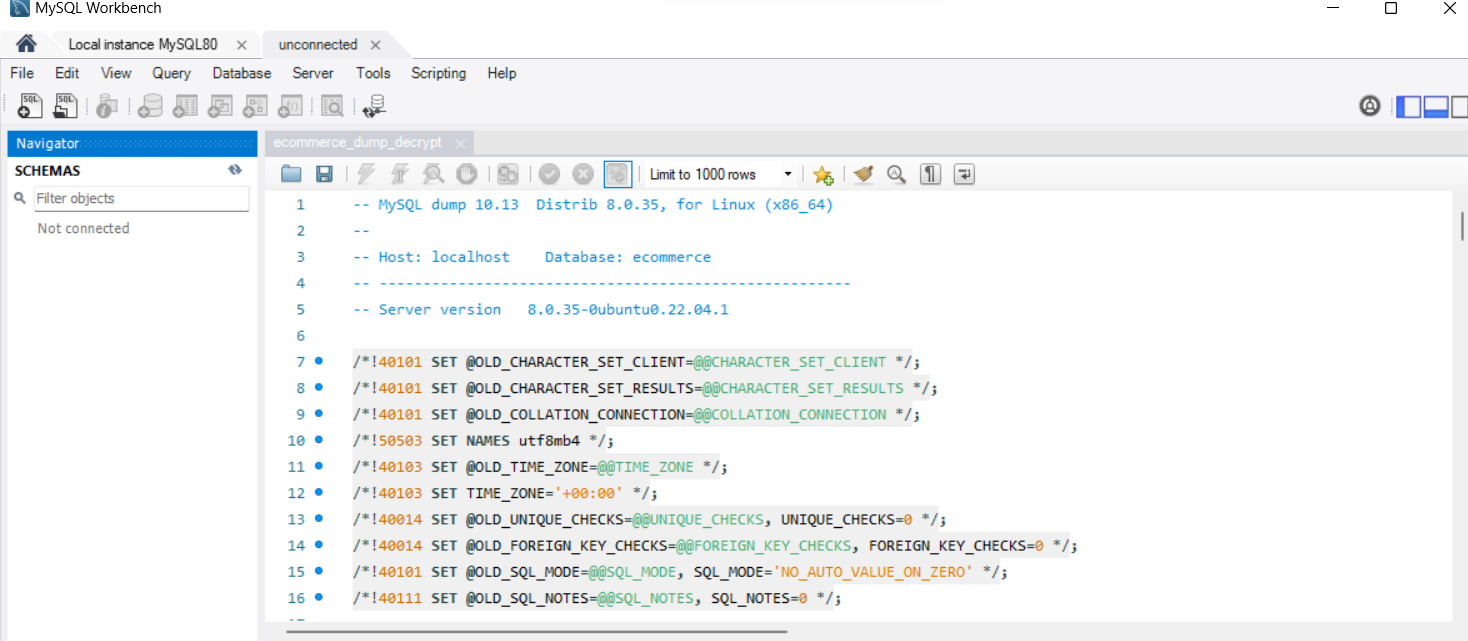
While performing decryption, the system will ask for the passphrase as shown below, This passphrase was while generating gnupg key.



This is the command for decryption using gnupg



As the file was seen before the encryption, all the table structures were visible. Similarly after decryption the shown encrypted file, all the table structures will be visible as shown below.



**For data in transit:**

Data encryption is a fundamental aspect of securing sensitive information, especially when it is in transit. The use of GnuPG (GNU Privacy Guard) provides a robust and widely adopted method for implementing encryption. GnuPG employs asymmetric key cryptography, offering a secure means to encrypt and decrypt data. Here's a theory on implementing data encryption using GnuPG for data in transit:

Background Information:

In the context of an e-commerce database, securing data in transit is crucial to prevent unauthorized access during communication between different components of the system, such as the application server and the database server. GnuPG, an open-source implementation of the OpenPGP standard, offers a reliable solution for encrypting data in a way that ensures confidentiality and integrity.

Steps to Implement Data Encryption using GnuPG:

Install GnuPG:

* Ensure that GnuPG is installed on both the sender and receiver systems. GnuPG is available for various operating systems, and installation instructions can be found on the official GnuPG website or through package managers.

sudo apt-get update

sudo apt-get install gnupg

Generate Key Pairs:

* On both the sender and receiver sides, generate a key pair using GnuPG. The key pair consists of a public key (used for encryption) and a private key (used for decryption). Each user or system involved in the data exchange should have its own key pair.

gpg --gen-key

Key Exchange:

* Share public keys securely between the sender and receiver. This is a critical step in asymmetric encryption, where each entity uses the other's public key to encrypt data. Various secure methods, such as secure file transfer or key servers, can be employed for key exchange.

gpg --export -a "Your Name" > public\_key.asc

Encrypt Data:

* Before transmitting sensitive data, encrypt it using the recipient's public key. GnuPG provides a simple command-line interface for encryption. For example:

gpg --encrypt --recipient recipient@example.com file\_to\_encrypt.txt

Replace "recipient@example.com" with the email address associated with the recipient's public key.

Transmit Encrypted Data:

* Transmit the encrypted data securely over the network. Common protocols for secure transmission include HTTPS, SFTP, or any other encrypted communication channel that suits your specific environment.
* Transmit the encrypted file (**file\_to\_encrypt.txt.gpg**) securely to the receiver.

Receiver's System:

Install dependencies such as gnupg and update the system.

Import public key:

* gpg --import public\_key.asc
* Ensure that the **public\_key.asc** file received from the sender is in the current directory.

Decrypt Received Data:

* gpg --decrypt file\_to\_encrypt.txt.gpg

These steps will allow to encrypt and decrypt the data in transit form the database.

**Task 3: Project Management Components**

**Project Management Plan (PMP)**

Created PMP plan for this project and attached a separate document in the submission.

Attachments include: PMP file and EVS analysis file.

**Task 4: Presentation and Documentation**

Created Power Point Presentation for this project and attached a separate document in the submission.

Attachments include: PPT file.