**Part 1 - To Remain with the Assignment after Marking**

|  |  |
| --- | --- |
| **Student ID: E11261** | **Student Name: Belpa Gamage Ranuga Disansa** |
| **Module Code: CI6320** | **Module Name: Advanced Data Modelling** |
| **Assignment number: 1** | **ESoft Module Leader: Mr. W A D B C Goonatillaka** |
| **Date set: 16th of March 2024** | **Date due: 7th of April 2024** |

**Guidelines for the Submission of Coursework**

1. Print this cover sheet and securely attach both pages to your assignment. You can help us ensure work is marked more quickly by submitting at the specified location for your module. You are advised to keep a copy of every assignment.

2. Coursework deadlines are strictly enforced by the University.

3. You should not leave the handing in of work until the last minute. Once an assignment has been submitted it cannot be submitted again.

**Academic Misconduct**: **Plagiarism** and/or **collusion** constitute **academic misconduct** under the University's Academic Regulations. Examples of academic misconduct in coursework: making available your work to other students; presenting work produced in collaboration with other students as your own (unless an explicit assessment requirement); submitting work, taken from sources that are not properly referenced, as your own. By printing and submitting this coversheet with your coursework you are confirming that the work is your own.

|  |  |  |  |
| --- | --- | --- | --- |
| |  | | --- | | ESoft Office Use Only:  Date stamp: work received. | | |  | | --- | |  | |

**Part 2 – Student Feedback**

|  |  |
| --- | --- |
| **Student ID:** | **Student Name:** |
| **Module Code:** | **Module Name:** |
| **Assignment number:** | **ESoft Module Leader:** |
| **Date set:** | **Date due:** |

|  |
| --- |
| Strengths (areas with well-developed answers) |

|  |
| --- |
| Weaknesses (areas with room for improvement) |

|  |
| --- |
| Additional Comments |

|  |  |  |
| --- | --- | --- |
|  |  |  |
| **ESoft Module Lecturer:** | **Provisional mark as %:** |  |
| **ESoft Module Marker:** | **Date marked:** |

Acknowledgement

I want to express my appreciation to those who have helped complete this report.

I have been able to complete this report because of the Academic advisor Mr. W A D B C Goonatillaka for their invaluable support, mentorship, and feedback and the faculty members of CI6320(Advanced Data Modelling).

Additionally, I would like to thank my family members for their unwilling encouragement and help throughout the report.

Thank you.

Table of Contents

[Part A 7](#_Toc163561360)

[1. Introduction 7](#_Toc163561361)

[1.1 What is data modelling? 7](#_Toc163561362)

[1.2 Importance of Data Models 8](#_Toc163561363)

[2. Relational Data Model (RDM) 8](#_Toc163561364)

[2.1 History 8](#_Toc163561365)

[2.2 Core Principles 8](#_Toc163561366)

[2.3 Characteristics 8](#_Toc163561367)

[3. Object-Oriented Data Model (OODM) 8](#_Toc163561368)

[3.1 History 8](#_Toc163561369)

[3.2 Core Principles 8](#_Toc163561370)

[3.3 Characteristics 9](#_Toc163561371)

[4. Object-Relational Data Model (ORDM) 9](#_Toc163561372)

[4.1 History 9](#_Toc163561373)

[4.2 Core Principles 9](#_Toc163561374)

[4.3 Characteristics 9](#_Toc163561375)

[5. Summary 9](#_Toc163561376)

[5.1 Comparison Table 9](#_Toc163561377)

[5.2 Critical Discussion on Which Model to use in real world application scenarios. 10](#_Toc163561378)

[Part B 10](#_Toc163561379)

[1. Setting Up the Development Environment 10](#_Toc163561380)

[2. Publication 12](#_Toc163561381)

[2.1 Type Creation 12](#_Toc163561382)

[2.2 Body Creation 13](#_Toc163561383)

[2.3 Table Creation 14](#_Toc163561384)

[2.4 Data Insertion 15](#_Toc163561385)

[2.5 Data Retrieval 16](#_Toc163561386)

[3. Book 17](#_Toc163561387)

[3.1 Type Creation 17](#_Toc163561388)

[3.2 Body Creation 18](#_Toc163561389)

[3.3 Table Creation 19](#_Toc163561390)

[3.4 Data Insertion 20](#_Toc163561391)

[4. Journal 21](#_Toc163561392)

[4.1 Type Creation 21](#_Toc163561393)

[4.2 Body Creation 22](#_Toc163561394)

[4.3 Table Creation 22](#_Toc163561395)

[4.4 Data Insertion 23](#_Toc163561396)

[5. Member 24](#_Toc163561397)

[5.1 Type Creation 24](#_Toc163561398)

[5.2 Body Creation 25](#_Toc163561399)

[5.3 Table Creation 26](#_Toc163561400)

[5.4 Data Insertion 27](#_Toc163561401)

[6. Loans 28](#_Toc163561402)

[6.1 Table Creation 28](#_Toc163561403)

[6.2 Data Insertion 29](#_Toc163561404)

[7. Additional Features 30](#_Toc163561405)

[7.1 Additional attributes for Publications 30](#_Toc163561406)

[7.2 Loan Details Viewer 31](#_Toc163561407)

[References 33](#_Toc163561408)

Table of Figures

[Figure 1Oracle Database Version 10](#_Toc163561110)

[Figure 2Creating / Replacing Type Publication Execution 11](#_Toc163561111)

[Figure 3Creating / Replacing Type Publication SQL 11](#_Toc163561112)

[Figure 4Creating / Replacing Publication Type Body Execution 12](#_Toc163561113)

[Figure 5Creating / Replacing Publication Body SQL 12](#_Toc163561114)

[Figure 6Creating Publications Table Execution 13](#_Toc163561115)

[Figure 7Creating Publications Table SQL 13](#_Toc163561116)

[Figure 8Inserting Data into the Publications Table Execution 14](#_Toc163561117)

[Figure 9Inserting into the Publications Table SQL 14](#_Toc163561118)

[Figure 10Retriving Data from Publications Table Execution 15](#_Toc163561119)

[Figure 11Retriving Data from Publications Table SQL 15](#_Toc163561120)

[Figure 12Alternating the Type Publications to be extended. 16](#_Toc163561121)

[Figure 13Creating Type Book Execution 16](#_Toc163561122)

[Figure 14Creating Type Book SQL 16](#_Toc163561123)

[Figure 15Creating / Replacing Body Type Execution 17](#_Toc163561124)

[Figure 16Creating / Replacing Body Type SQL 17](#_Toc163561125)

[Figure 17Create Table Books Execution 18](#_Toc163561126)

[Figure 18Create Table Books SQL 18](#_Toc163561127)

[Figure 19Inserting into the Books Table Execution 19](#_Toc163561128)

[Figure 20Insering into the Books Table SQL 19](#_Toc163561129)

[Figure 21 Alternating the Type Publications to be extended. 20](#_Toc163561130)

[Figure 22Creating Type Journal Execution 20](#_Toc163561131)

[Figure 23Creating Type Journal SQL 21](#_Toc163561132)

[Figure 24Creating / Replacing Journal Type Body Execution 21](#_Toc163561133)

[Figure 25Creating / Replacing Journal Type Bodyy SQL 21](#_Toc163561134)

[Figure 26Creating Journals Table Execution 22](#_Toc163561135)

[Figure 27Creating / Replacing Journal Table SQL 22](#_Toc163561136)

[Figure 28Inserting Data into the Journals Table Execution 22](#_Toc163561137)

[Figure 29Creating Type Memeber Execution 23](#_Toc163561138)

[Figure 30Creating Type Member SQL 23](#_Toc163561139)

[Figure 31Creating / Replacing Member Type Body Execution 24](#_Toc163561140)

[Figure 32Creating / Replacing Member Type Body SQL 24](#_Toc163561141)

[Figure 33Creating Members Table Execution 25](#_Toc163561142)

[Figure 34Creating Memebers Table SQL 25](#_Toc163561143)

[Figure 35Inserting Data into the Members Table Execution 26](#_Toc163561144)

[Figure 36Inserting Data into the Members Table SQL 26](#_Toc163561145)

[Figure 37Creating Table Loans Execution 27](#_Toc163561146)

[Figure 38Creating Table Loans SQL 27](#_Toc163561147)

[Figure 39Inserting Data into the Loans Table Execution 28](#_Toc163561148)

[Figure 40Inserting Data into the Loans Table SQL 28](#_Toc163561149)

[Figure 41Altering Publications Table Execution 29](#_Toc163561150)

[Figure 42Altering Publications Table SQL 29](#_Toc163561151)

[Figure 43Loan Details Viewer Execution 30](#_Toc163561152)

[Figure 44Loan Details Viewer SQL 31](#_Toc163561153)

# Part A

## 1. Introduction

### What is data modelling?

A data model is a simple visual blueprint for designing a database or software system—depicting data entities, attributes, and their relationships. It helps businesses organize their data effectively, design or re-engineer databases, and align with business and application requirements. A data model aids in creating a bridge between business and technical teams by turning real-world objects into a structure for a database. A data model is a critical first step after defining business requirements. (IBM, no date; Staff, 2023; G, 2018; DASCA, no date)

### Importance of Data Models

* Documentation
* Ensure data integrity.
* Higher Quality
* Decision Making

## 2. Relational Data Model (RDM)

### 2.1 History

The relational data model was introduced by E.F. Codd at IBM in San Jose, where a new data representation framework called the relational data model was established. It suggested that all the data could be stored in a tabular structure, in turn leading to higher productivity in the early 1980s (Newcomb and Couch, 2010). It is taken into consideration as the landmark or the start of database systems. (Khan, no date)

### 2.2 Core Principles

The RDM Model’s main principle is that data is organized into tables with columns and rows. Every table has a relation, and each row could be considered an instance of a relation. The columns are defined as attributes that are characteristics of the data. The model is a computer representation of mathematical theories of set theory and predicate logic. Relationships are used to store information about objects in a database. (Khan, no date.)

### 2.3 Characteristics

* Table based structure.
* Atomic values
* Normalization
* SQL

## 3. Object-Oriented Data Model (OODM)

### 3.1 History

The object-oriented data model (OODM) was developed to define operations for designing schemas, creating databases, retrieving objects, and navigating while supporting features such as aggregation, generalization, and particularization relationships (Zhao, 1988).

### 3.2 Core Principles

The object-oriented data model represents the real world as objects with problems, attributes, and relationships. OODM was created by combining relational data model concepts with object-oriented programming principles (GFG, 2021). This approach allows classes to be grouped into items with comparable qualities, vacillating the organization and management of data structures while allowing for a smooth transition from the design concept to implementation in object-oriented databases (Janecatalla, 2012; Alzahrani, 2016).

### 3.3 Characteristics

* Objects are created from classes, which are like blueprints of the structure (GFG, 2021)
* Allows for inheritance allowing subclasses to inherit attributes and methods from existing classes (GFG 2021), which allows for code reuse.
* Operations are performed on the data that is encapsulated by the objects.

## 4. Object-Relational Data Model (ORDM)

### 4.1 History

With the limitations of both the relational and object-oriented data models, research in the 1990s led to the development of the object-oriented data model, which takes fundamental concepts from both the relational and object-oriented data model while addressing areas where improvement was sought. (Castro, 2020)

### 4.2 Core Principles

The Object-oriented model combines the important features of both the relational and object-oriented data models, and it has extracted important core principles such as Supporting objects, classes, and inheritance from the Object-oriented data model and data types and tables from the Relational Data Model (Castro, 2020; Auziņš, 2018).

### 4.3 Characteristics

* supports complex Data Types such as Arrays, Nested Tables, and user-defined types.
* Has object-oriented principles in combination with the features of the relational data model, in turn allowing for the creation of much more advanced objects with relational principles.
* Creates a data model which has the most important features to be able to model the real world while having the flexibility to represent complex relationships and structures.

## 5. Summary

### 5.1 Comparison Table

|  |  |  |  |
| --- | --- | --- | --- |
| Feature | Object Oriented Data Model (OODM) | Relational Data Model (RDM) | Object Relational Data Model (ORDM) |
| Data Representation | Objects with attributes and methods | Tables with rows and columns | Tables with rows, columns, and some OO concepts (inheritance, complex data types) |
| Relationships | Inheritance, Aggregation, and Association | Foreign Keys | Foreign Keys and some OO concepts (inheritance) |
| Performance | Potentially faster due to no joins | Can be efficient for specific queries | Can be efficient for specific queries but may be slower for complex relationships |
| Advantages | Code reuse (inheritance) and Semantic modelling easier to model complex relationships | flexible and efficient for certain queries Secure and Scalable | Combines benefits of OO and Relational models. Supports complex data types: Inheritance |
| Disadvantages | no strong mathematical foundation Difficulty with persistence for complex structures | can be complex to design for large data sets not ideal for complex querying | can be complex to manage; may not be as performant as pure OO for complex relationships |

### 5.2 Critical Discussion on Which Model to use in real world application scenarios.

Selecting a data model for real-world applications relies heavily on a comprehensive understanding of the specific requirements, constraints, and characteristics of the application.

Relational Data Models (RDM) are extremely effective in scenarios where data consistency and integrity are important, data for example a banking system and CRM Databases. Object-oriented models (OODM) have extremely rare use cases because they are useful in complex data modelling scenarios such as the natural representation of entities with a lot of behaviours and relationships in applications such as multimedia or gaming. Object Relational Data Models (ORDM) have a balance between flexibility and data integrity, making them extremely useful for hybrid scenarios such as social media platforms and e-commerce systems. The choice between the different data models requires careful evaluations of the advantages and disadvantages of each model, considering factors such as complexity, development, and requirements.

# Part B

## 1. Setting Up the Development Environment

To set up the development environment to execute the following given programs: First, make sure that Oracle Database and Developer are installed on the development machine. Then configure the connection with the Oracle database, and then it would be possible to replicate the following programs. The following attached figure shows the Oracle database version that was used in development.

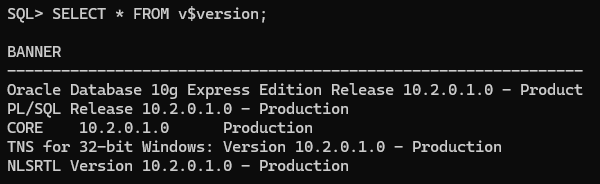


Figure 1Oracle Database Version

## 2. Publication

### Type Creation

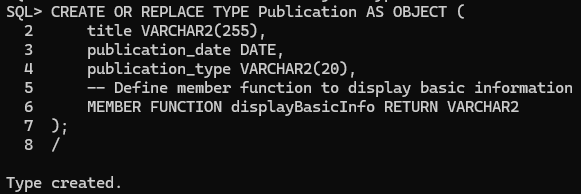


Figure 2Creating / Replacing Type Publication Execution

In this given SQL script creates or replaces a User Defined Type (UDT) called ‘Publication’. This type has 3 attributes which are: ‘title’ with VARCHAR (255) datatype, ‘publication\_date’ with DATE datatype and ‘publication\_type’ with VARCHAR2(2) datatype, and finally this defines a function called `displayBasicInfo()` which is expected to of type `VARCHAR2`.

A screenshot of a computer program

Description automatically generated

Figure 3Creating / Replacing Type Publication SQL

### Body Creation

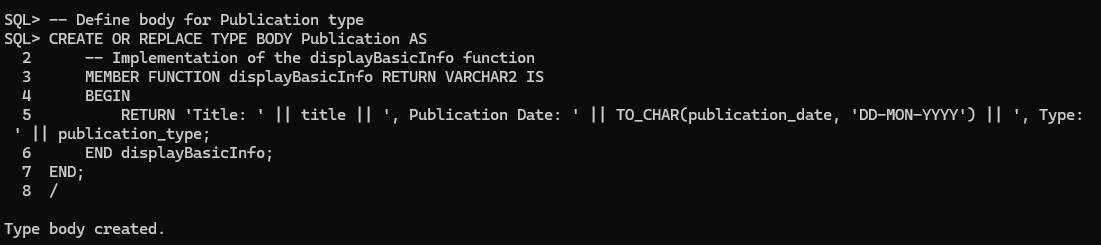


Figure 4Creating / Replacing Publication Type Body Execution

This block of SQL code defines the body type for the type `Publication`. It has the following components:

1. `CREATE OR REPLACE TYPE BODY Publication AS`: This statement either creates or replaces the Body for the Type Publication
2. `MEMBER FUNCTION displayBasicInfo RETURN VARCHAR2 IS`: This statement creates a member function called `displayBasicInfo()` which returns the data type of `VARCHAR2`.
3. `BEGIN`: This marks the start of the function implementation
4. `RETURN 'Title: ' || title || ', Publication Date: ' || TO\_CHAR(publication\_date, 'DD-MON-YYYY') || ', Type: ' || publication\_type;`: This SQL statement returns a concatenated string with details such as title, publication\_date and publication\_type
5. `END displayBasicInfo;`: Marks the end of the displayBasicInfo function definition.

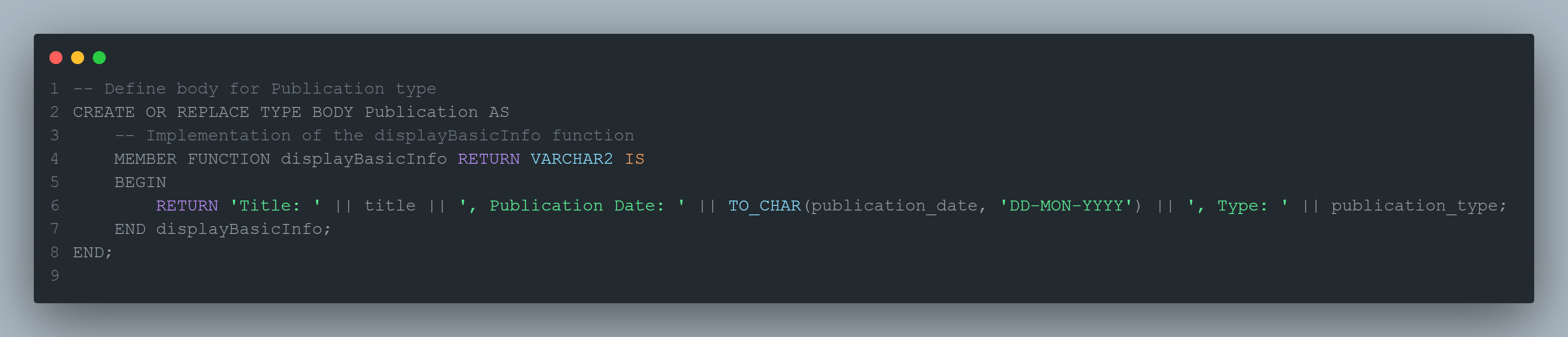


Figure 5Creating / Replacing Publication Body SQL

### Table Creation

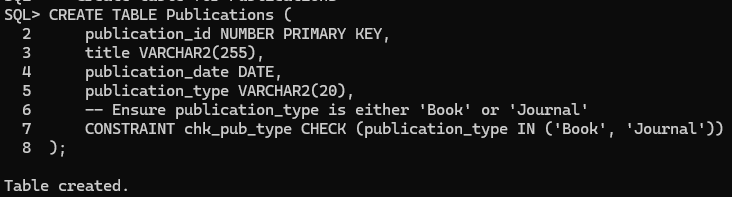


Figure 6Creating Publications Table Execution

The given SQL script creates a table named `Publications` with four columns: `publication\_id`, `title`, `publication\_date` and `publication\_type`. This table contains a constraint named `chk\_pub\_type` that ensures that the column will only contain the values `Book` or `Journal`.

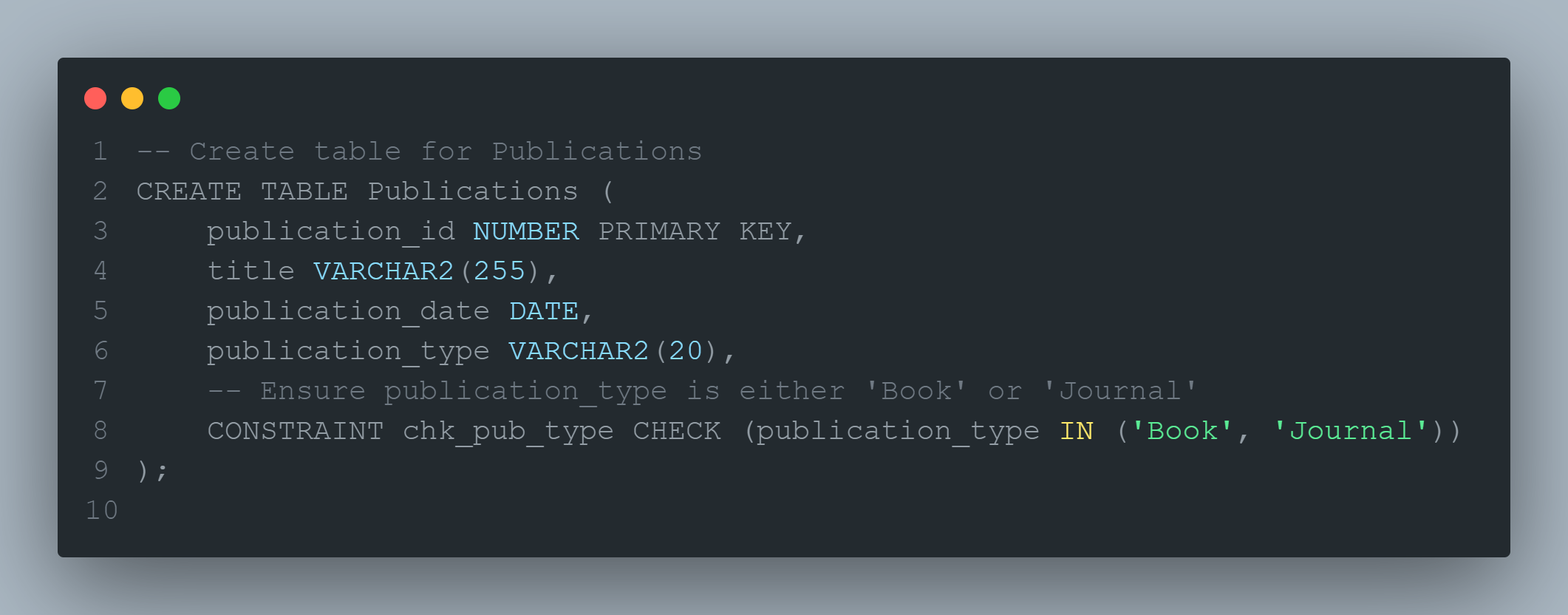


Figure 7Creating Publications Table SQL

### Data Insertion

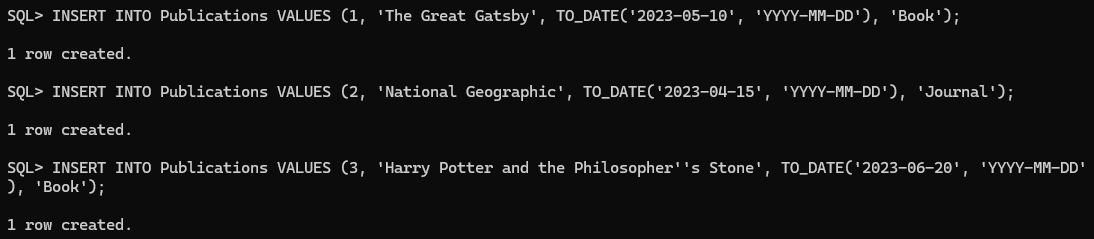


Figure 8Inserting Data into the Publications Table Execution

These 3 statements enter 3 records into the Publications Table, where each INSERT INTO inserts one row of data. Each value corresponds to a column in the table, and there are 4 columns, so each insertion has 4 values corresponding to each column. The TO\_DATE function is used to convert strings into the DATE data type/structure.



Figure 9Inserting into the Publications Table SQL

### Data Retrieval

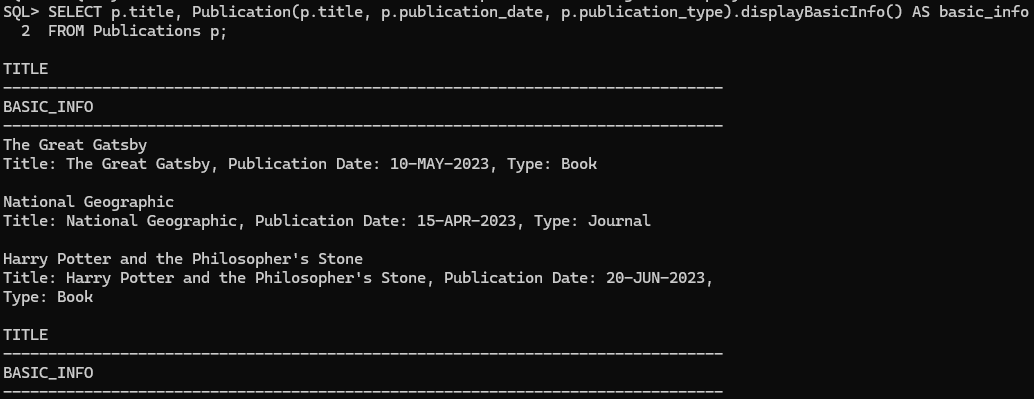


Figure 10Retriving Data from Publications Table Execution

This query selects data from the Publications table, and from each row, it retrieves its title column and uses the other attributes to create an instance of the `Publication` type and then calls the `displayBasicInfo` method on the `Publication` instance and the returned information is aliased as `basic\_info`, finally the title alongside the string from the `displayBasicInfo()`.

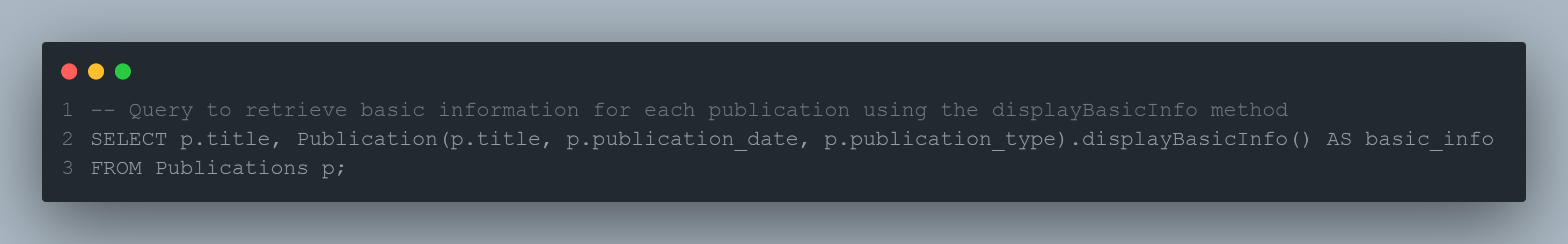


Figure 11Retriving Data from Publications Table SQL

## 3. Book

Before Executing any of the following SQL scripts the following SQL code must be executed to ensure that any errors won’t be raised.

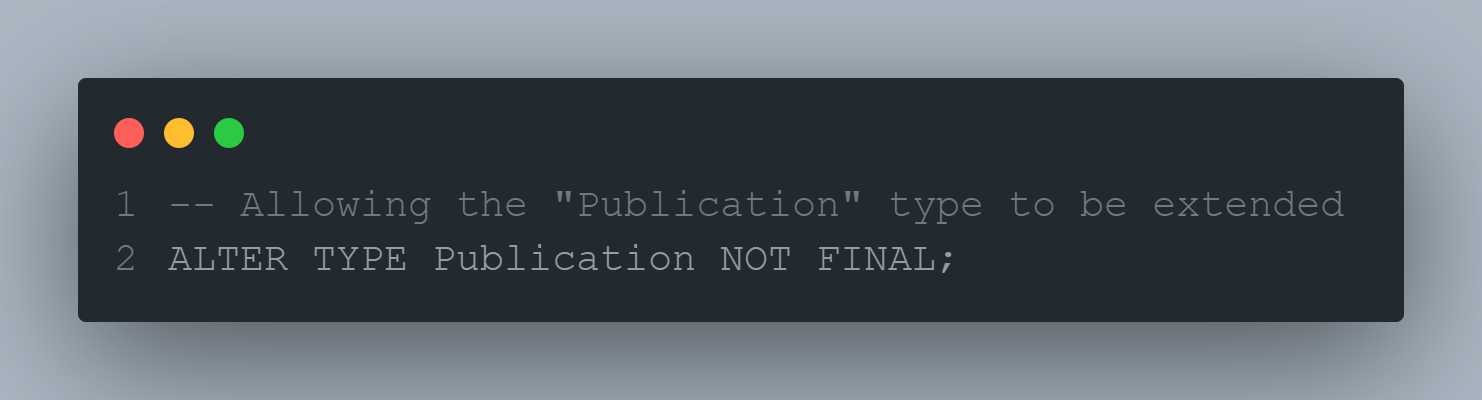


Figure 12Alternating the Type Publications to be extended.

### Type Creation

A black screen with white text

Description automatically generated

Figure 13Creating Type Book Execution

This code segment creates a subtype `Book` on top of the `Publication` supertype, it introduces more attributes such as `author` with VARCHAR2(255) datatype, `ISBN` with VARCHAR2(13) datatype; and the subtype introduces a new member function called `displayFullInfo()` which returns the data type `VARCHAR2`.

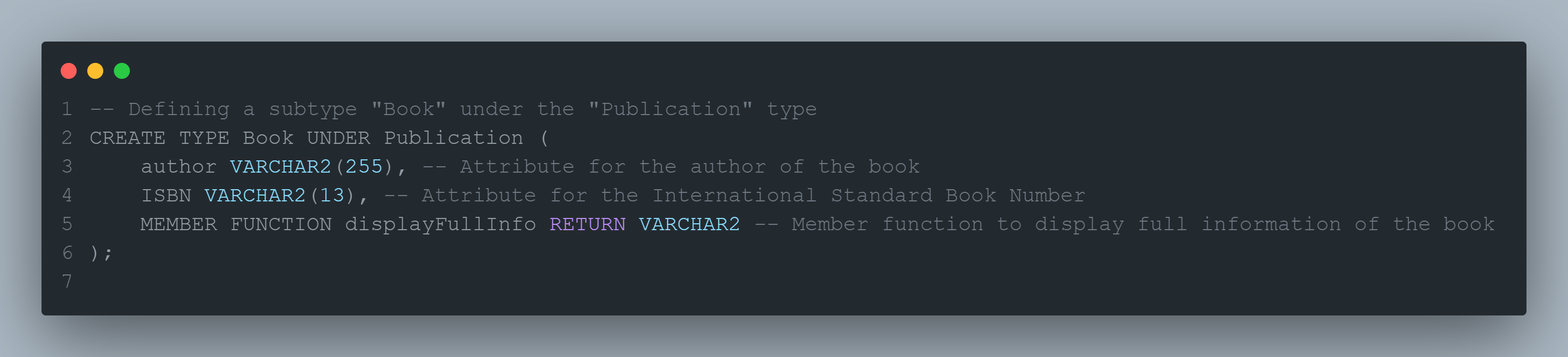


Figure 14Creating Type Book SQL

### Body Creation

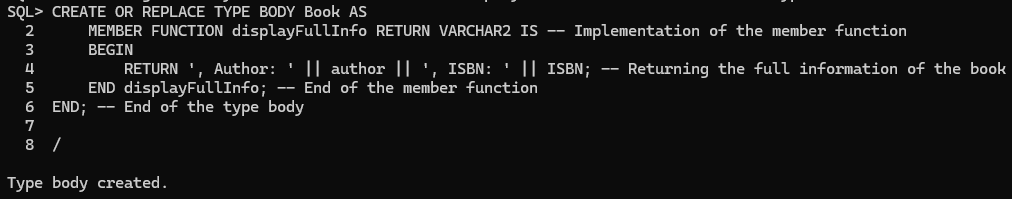


Figure 15Creating / Replacing Body Type Execution

This code implements the `displayFullInfo` member function belonging to the `Book` subtype. The function is implemented so that a concatenated string with the details of all the attributes of the Book instance is returned.

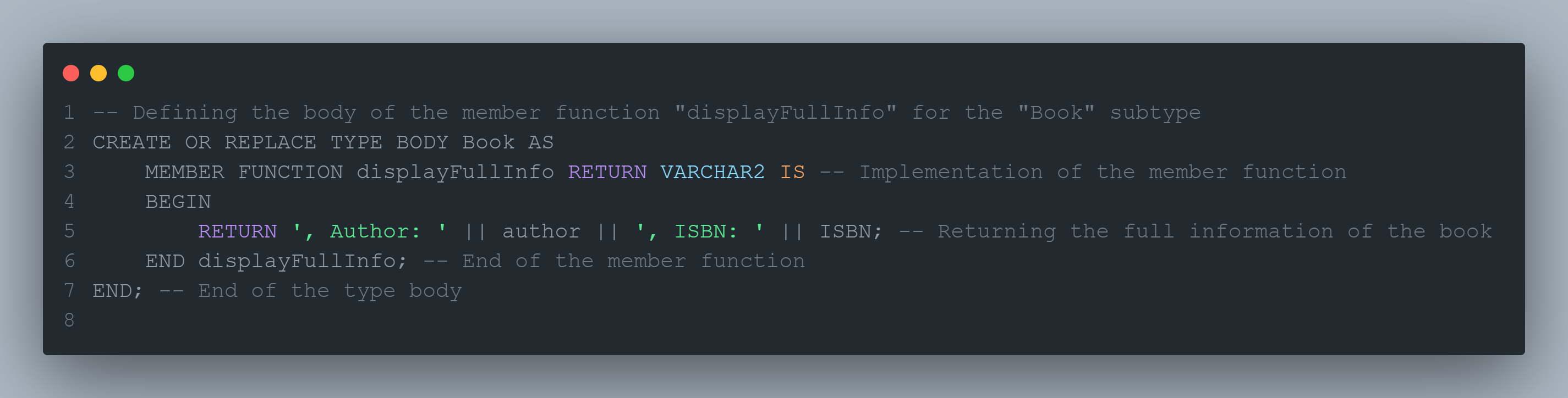


Figure 16Creating / Replacing Body Type SQL

### Table Creation

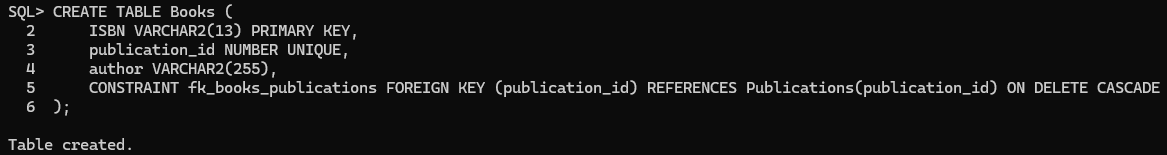


Figure 17Create Table Books Execution

This SQL code creates the table `Books`, which is used to store information regarding books. The table has 3 attributes and 1 constraint, The attributes are: ISBN with VARCHAR2(13) datatype, publication\_id with NUMBER datatype and author with VARCHAR2(255) datatype. The constraint in the Books table is that the `publication\_id` but be referenced in Publications(publication\_id) in turn creating a foreign key constraint.

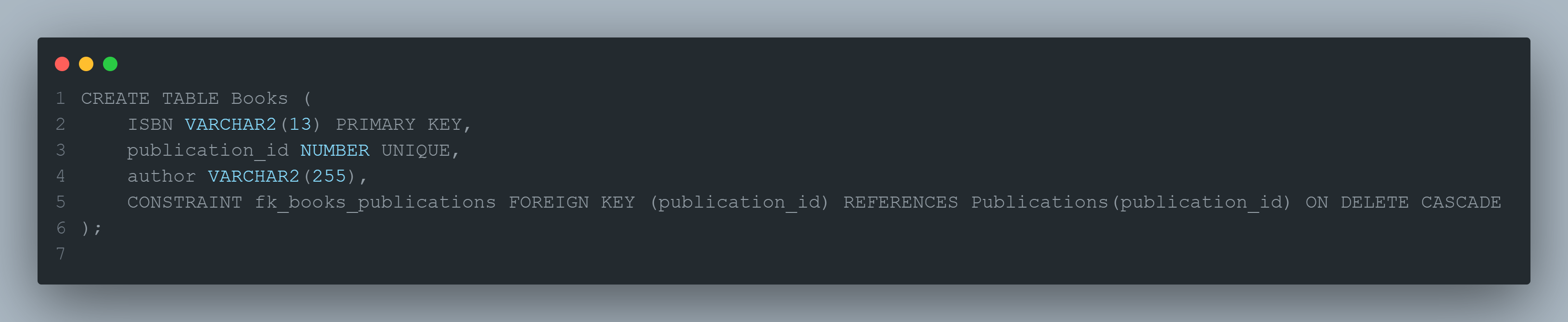


Figure 18Create Table Books SQL

### 3.4 Data Insertion

A computer screen shot of a black screen

Description automatically generated

Figure 19Inserting into the Books Table Execution

These 3 statements enter 3 records into the Books Table, where each INSERT INTO inserts one row of data. Each value corresponds to a column in the table, and there are 3 columns, so each insertion has 3 values corresponding to each column. Due to the constraint, all the `publication\_id`’s are already in the `Publications` table.

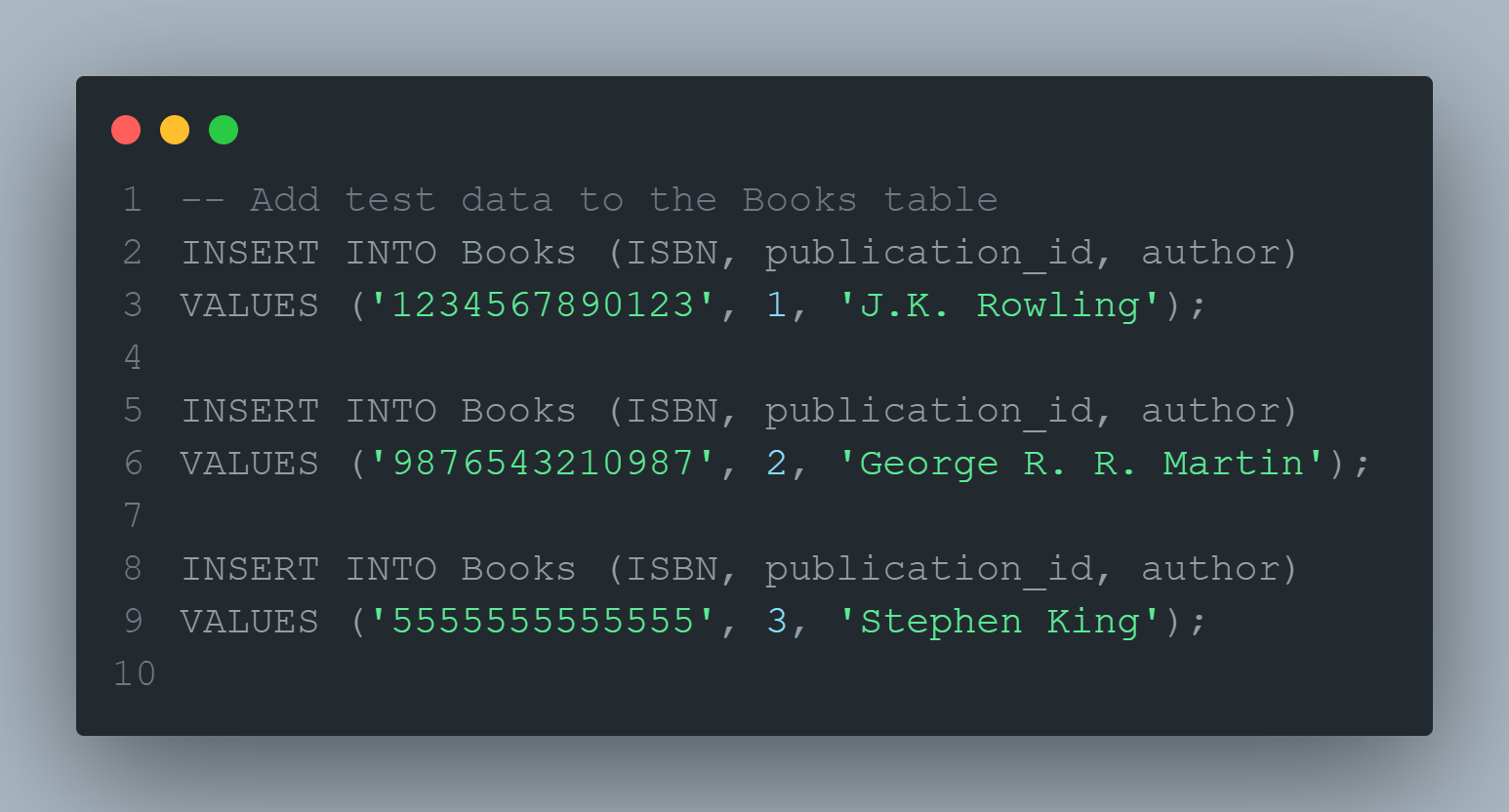


Figure 20Insering into the Books Table SQL

## 4. Journal

Before Executing any of the following SQL scripts the following SQL code must be executed to ensure that any errors won’t be raised.

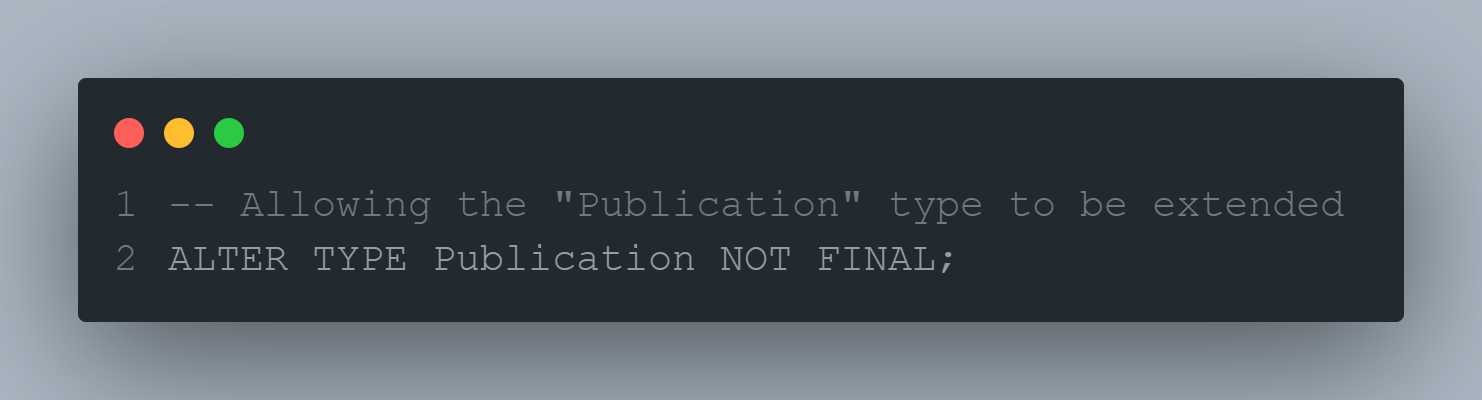


Figure 21 Alternating the Type Publications to be extended.

### 4.1 Type Creation

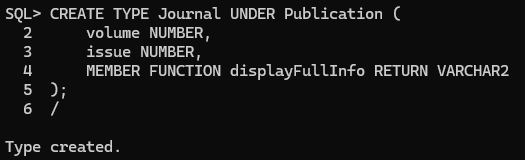


Figure 22Creating Type Journal Execution

This SQL statement creates a new object type, which is a subtype of the Publication supertype. It has 2 attributes and a member function called `displayFullInfo()` which returns the type of VARCHAR2.

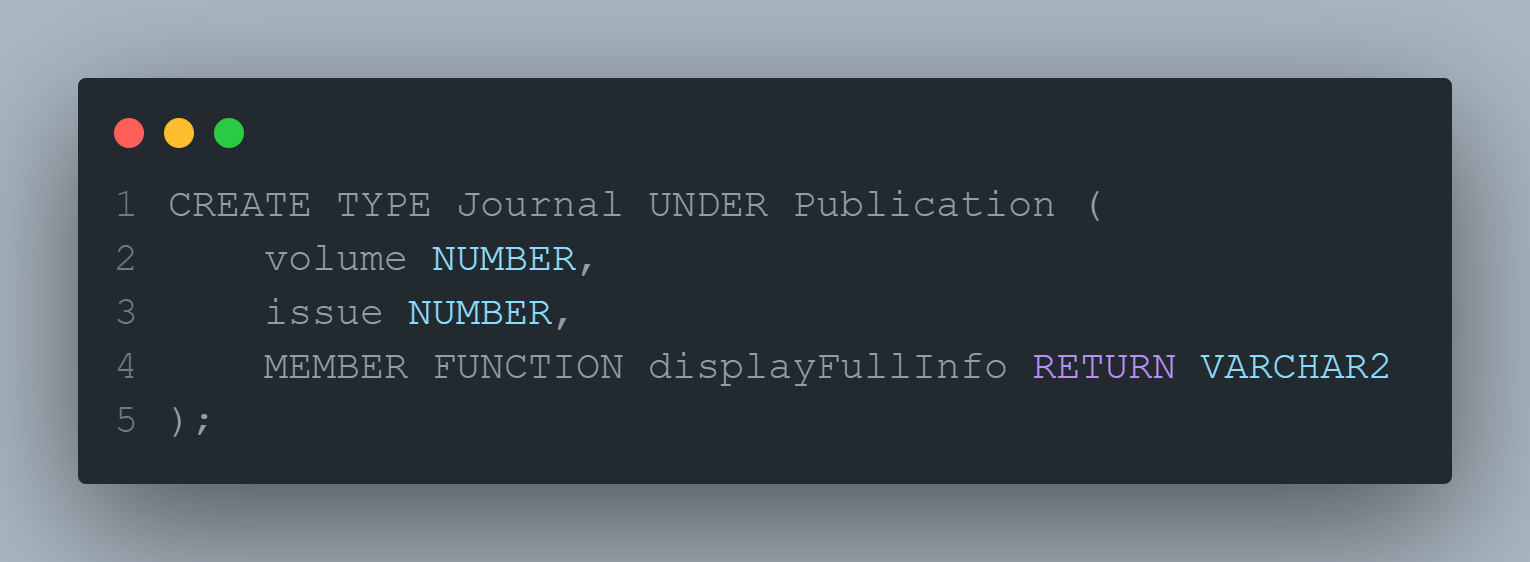


Figure 23Creating Type Journal SQL

### Body Creation

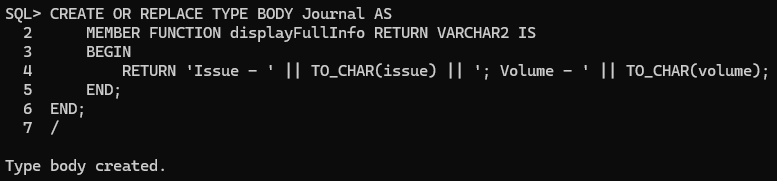


Figure 24Creating / Replacing Journal Type Body Execution

This code creates or replaces the body of the type `Journal` by implementing the function `displayFullInfo()` and it returns a string that contains all the attributes.

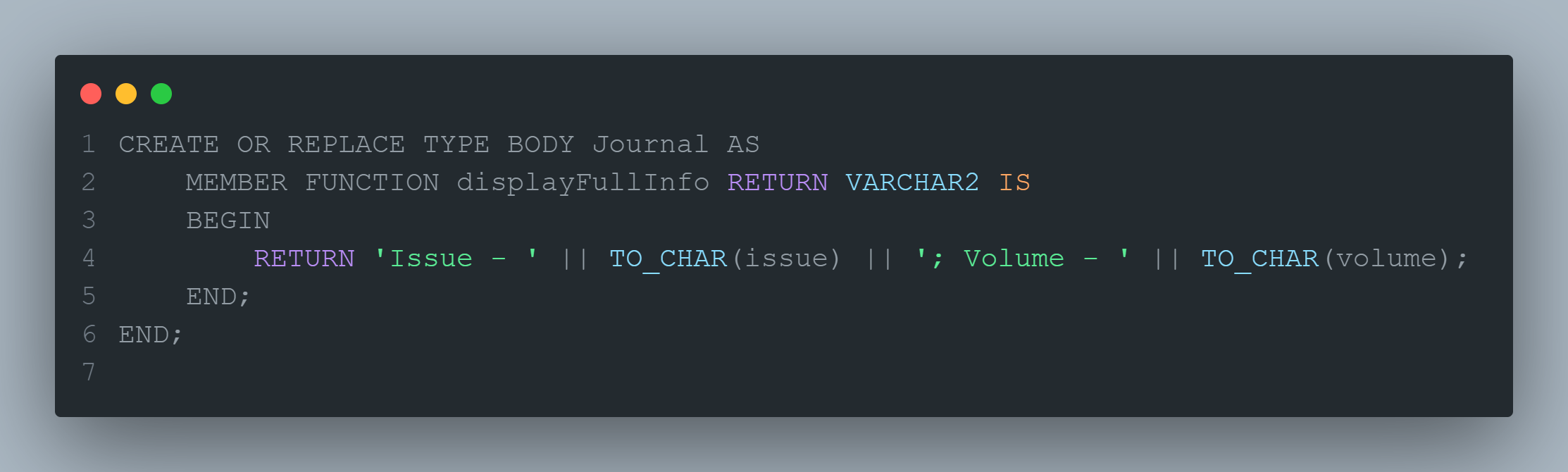


Figure 25Creating / Replacing Journal Type Bodyy SQL

### Table Creation

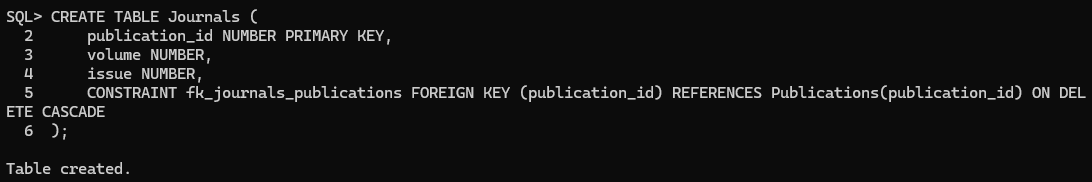


Figure 26Creating Journals Table Execution

This SQL Code creates the table `Journals`, which is used to store information regarding Journals. The table has 3 attributes and 1 constraint, the attributes are volume with NUMBER datatype, publication\_id with NUMBER datatype (which is the primary key) and issue with NUMBER datatype. The constraint in the Journals table is that the `publication\_id` references in Publications(publication\_id) in turn create a foreign key constraint.

Figure 27Creating / Replacing Journal Table SQL

### 4.4 Data Insertion

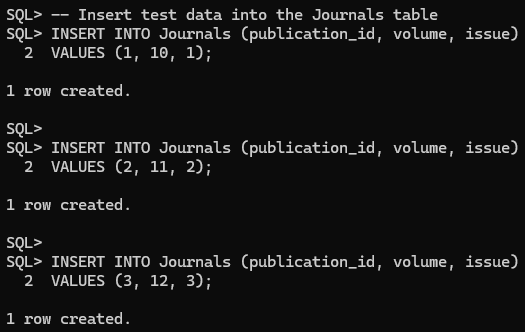


Figure 28Inserting Data into the Journals Table Execution

These 3 statements enter 3 records into the Journals Table, where each INSERT INTO inserts one row of data. Each value corresponds to a column in the table, and there are 3 columns, so each insertion has 3 values corresponding to each column. Due to the constraint, all the `publication\_id`’s are already in the `Publications` table.

# 5. Member

### 5.1 Type Creation

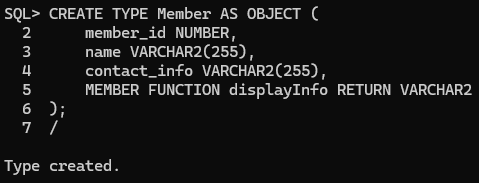


Figure 29Creating Type Memeber Execution

This SQL statement creates a new object type Member, it has 3 attributes (member\_id, name, contact\_info) and a member function called `displayFullInfo()` which returns the type of VARCHAR2.

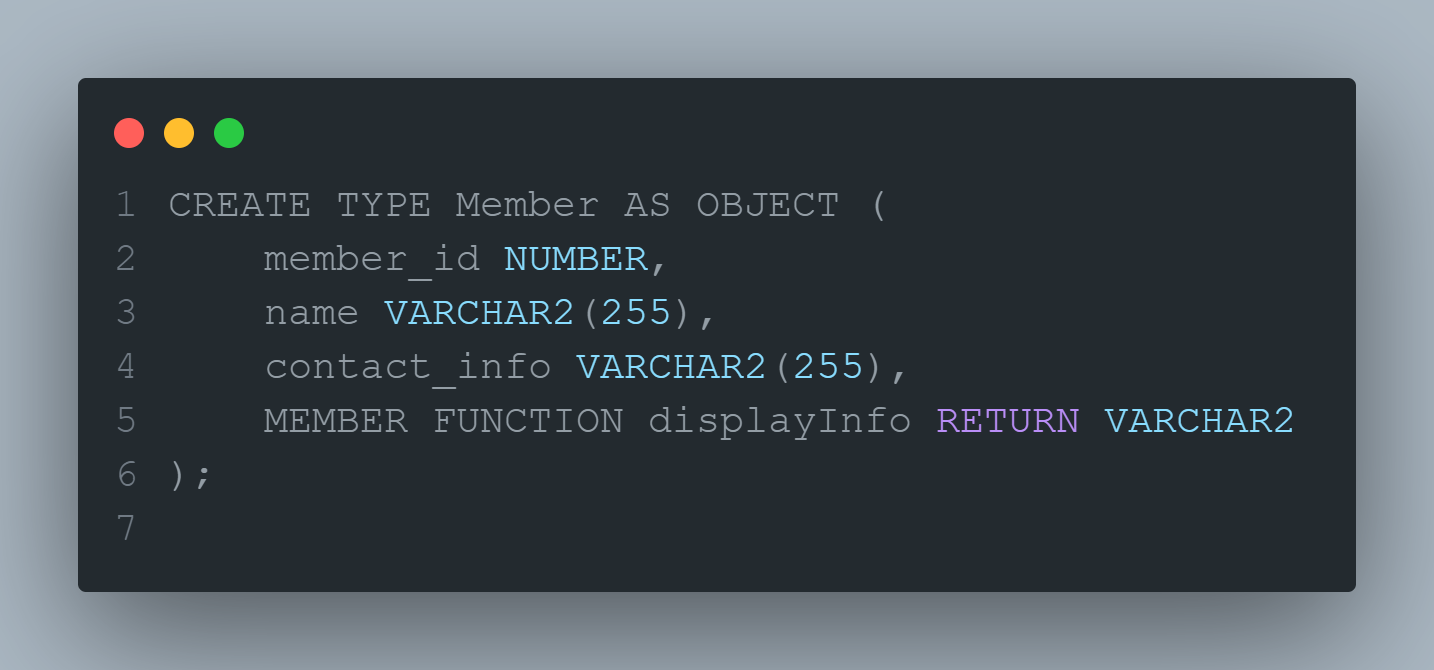


Figure 30Creating Type Member SQL

### Body Creation

A black screen with white text

Description automatically generated

Figure 31Creating / Replacing Member Type Body Execution

This code creates or replaces the body of the type `Member` by implementing the function `displayFullInfo()` and it returns a string that contains all the attributes.

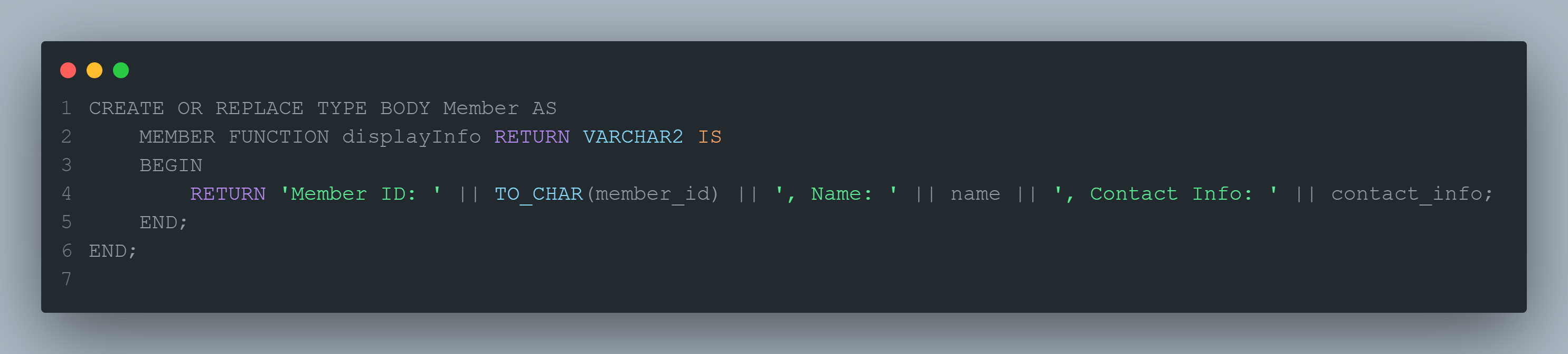


Figure 32Creating / Replacing Member Type Body SQL

### Table Creation

A black screen with white text

Description automatically generated

Figure 33Creating Members Table Execution

This SQL Code creates the table `Members`, which is used to store information regarding Members. The table has 3 attributes, the attributes are: member\_id with NUMBER datatype (which is the primary key), name with VARCHAR2(255) datatype and contact\_info with VARCHAR(255) datatype.

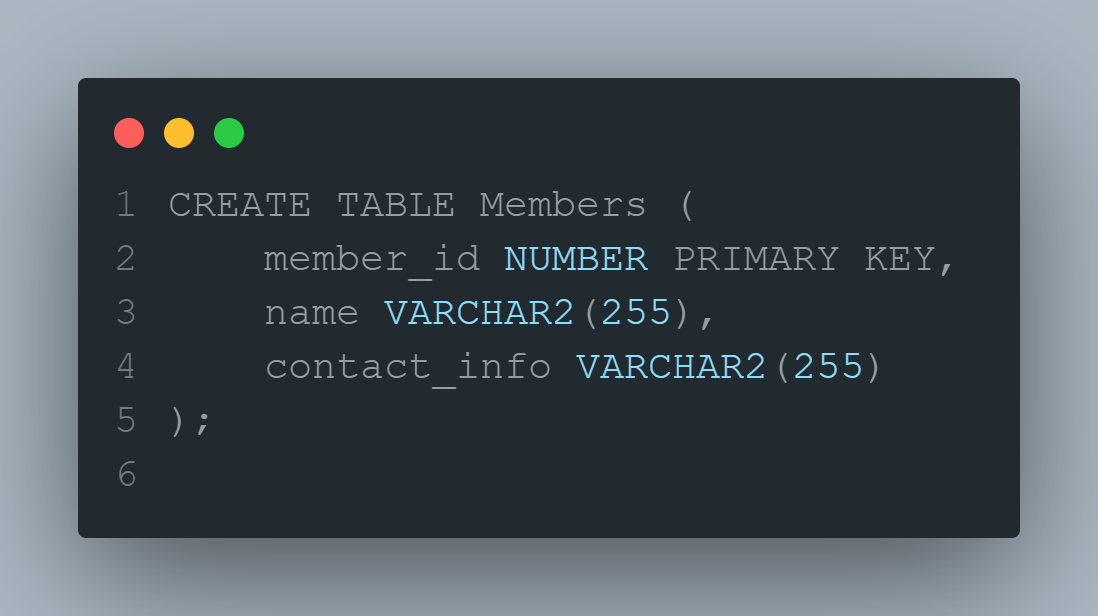


Figure 34Creating Memebers Table SQL

### 5.4 Data Insertion

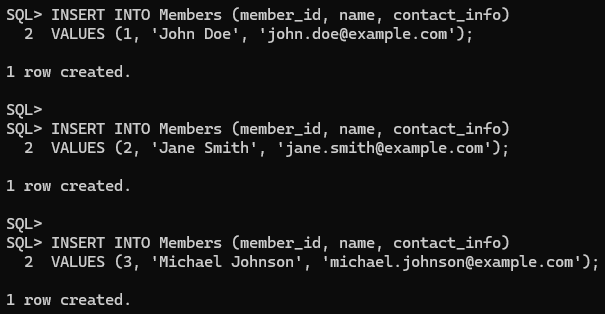


Figure 35Inserting Data into the Members Table Execution

These 3 statements enter 3 records into the Members Table, where each INSERT INTO inserts one row of data. Each value corresponds to a column in the Table, and there are 3 columns, so each insertion has 3 values corresponding to each column.



Figure 36Inserting Data into the Members Table SQL

## 6. Loans

To execute the following SQL code it is required that the above SQL code has already been executed.

### Table Creation

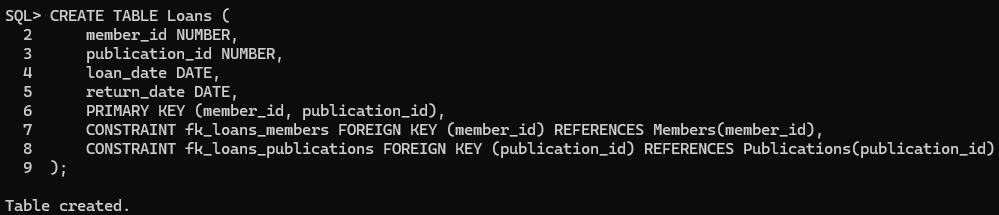


Figure 37Creating Table Loans Execution

This SQL statement creates a new object type Loans, it has 4 attributes: member\_id, publication\_id (which both are primary keys), loan\_date, and return\_date. There are 2 constraints in the table, 1 which make sure that the member\_id exists in the Members (member\_id) attribute, and another to make sure that the `publication\_id` exists in the Publications(publication\_id).

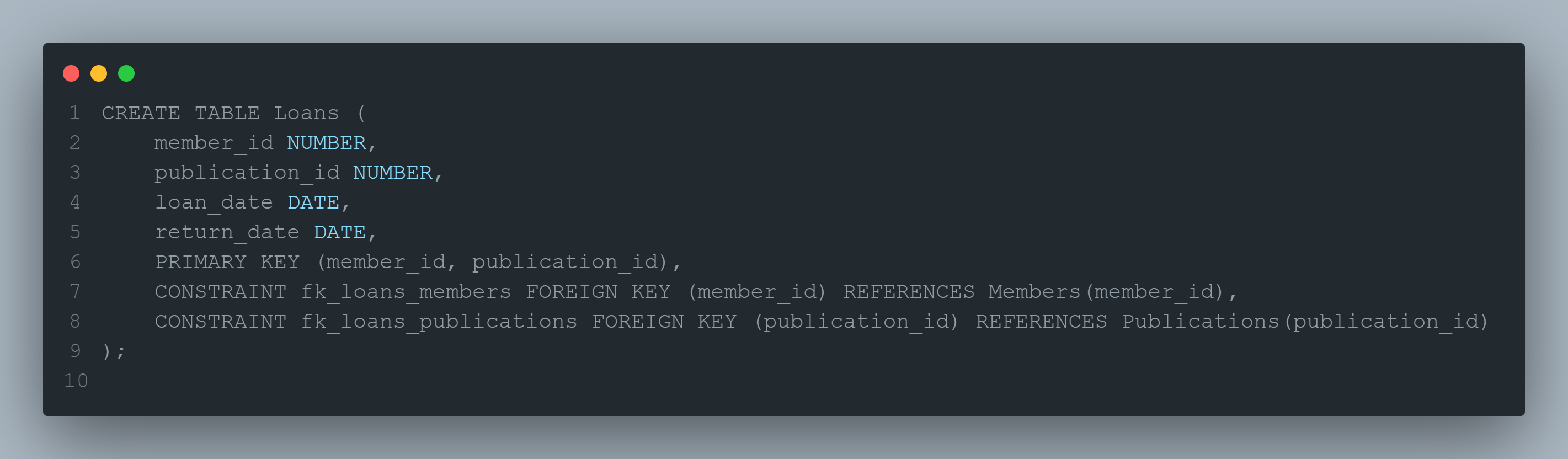


Figure 38Creating Table Loans SQL

### 6.2 Data Insertion

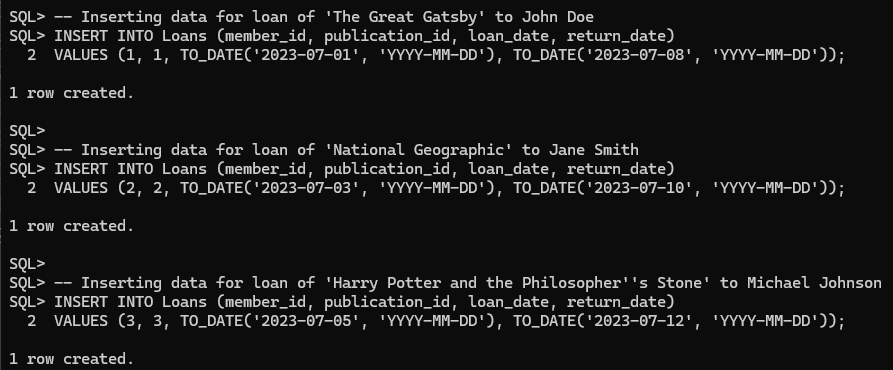


Figure 39Inserting Data into the Loans Table Execution

These 3 statements enter 3 records into the Loans Table, where each INSERT INTO inserts one row of data. Each value corresponds to a column in the Table, and there are 4 columns, so each insertion has 4 values corresponding to each column.

A computer screen shot of text

Description automatically generated

Figure 40Inserting Data into the Loans Table SQL

## 7. Additional Features

### 7.1 Additional attributes for Publications

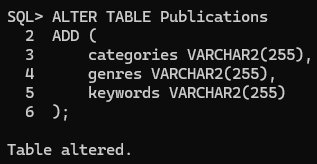


Figure 41Altering Publications Table Execution

The given SQL segment alters the table Publications to include 3 new columns / attributes: categories, genres, and keywords and all of them are of type VARCHAR2 with maximum characters of 255.

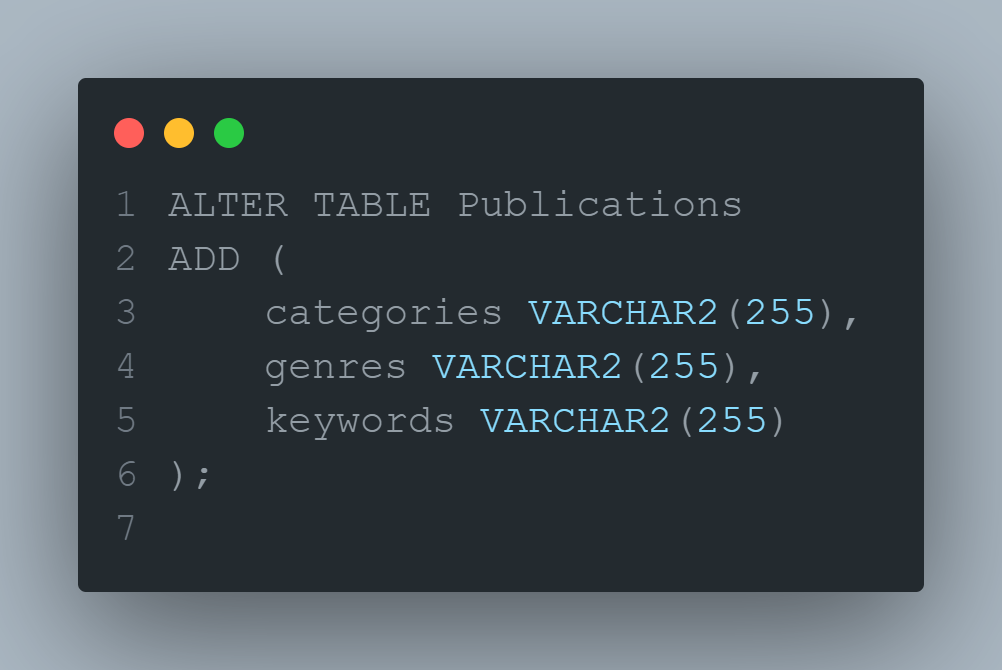


Figure 42Altering Publications Table SQL

### 7.2 Loan Details Viewer



Figure 43Loan Details Viewer Execution

This code creates a view named LoanDetails which collects data from Tables: Loans, Members and Publications and displays details such as member and publication details, loan dates in turn making it easier to analyze loan related information.

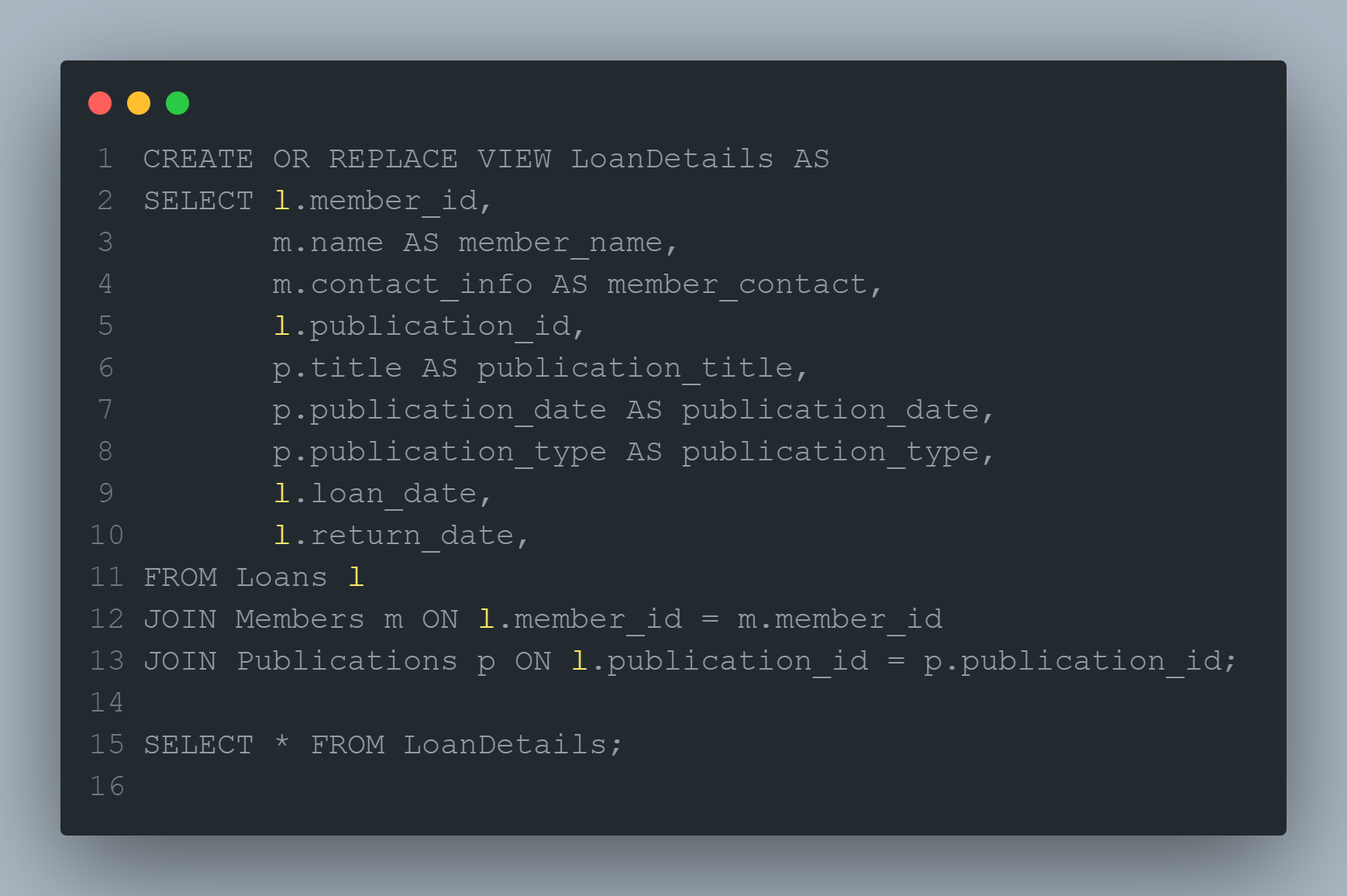


Figure 44Loan Details Viewer SQL

# References

Ameya (2024) 'What is relational Database with Real-Life examples,' RedSwitches, 31 January. <https://www.redswitches.com/blog/what-is-relational-database/>.

GfG (2021) Basic object oriented data model. <https://www.geeksforgeeks.org/basic-object-oriented-data-model/>.

*Object-Oriented Data Model and its Application | Free essay example* (2022). <https://studycorgi.com/object-oriented-data-model-and-its-application/>.

Bhagwat, S. (2022) *What is Object Oriented Model in DBMS? - Scaler Topics*. <https://www.scaler.com/topics/object-oriented-model-in-dbms/>.

Dancuk, M. (2023) What is an Object-Oriented Database. <https://phoenixnap.com/kb/object-oriented-database>.

*Object-relational data model* (no date). <https://www.tutorialspoint.com/Object-relational-Data-Model>.

Auziņš, A. *et al.* (2018) 'Object-Relational database structure model and structure optimisation,' *Applied Computer Systems (Online)*, 23(1), pp. 28–36. <https://doi.org/10.2478/acss-2018-0004>.

Jordan, M. (2023) *What is Database Scalability? Definition & FAQs | ScyllaDB*. <https://www.scylladb.com/glossary/database-scalability/#:~:text=Are%20relational%20databases%20scalable%3F,a%20vertical%20approach%20to%20scaling>.

Murphy, E. (2022) 'Flexible Relational Data,' *induro*, 13 May. <https://induro.io/blog/2022/05/flexible_relational_data/#:~:text=When%20relationships%20are%20predictable%20and,flexible%20to%20accurately%20represent%20it>.

Ot, A. (2023) *Relational Data Model 101: Key components & benefits*. <https://www.datamation.com/big-data/relational-data-model/#:~:text=Relational%20data%20models%20also%20aid,data%20sets%20in%20various%20applications>.

Pahi, N. (2013) *The relational model and normalization*. <https://blog.oureducation.in/relational-model-normalization/>.

*What is data modeling? | IBM* (no date b). <https://www.ibm.com/topics/data-modeling>.

*What is a Data Model?* (no date b). <https://cedar.princeton.edu/understanding-data/what-data-model>.

Staff, C. (2023b) *What is a data model?* <https://www.coursera.org/articles/data-model>.

G, M. (2018b) *Why Data Modelling is Important*. <https://www.linkedin.com/pulse/why-data-modelling-important-munish-goswami/>.

*What is Data Modeling and Why Do You Need It?* (no date b). <https://www.dasca.org/world-of-big-data/article/what-is-data-modeling-and-why-do-you-need-it>.

Vivek, J. (2024b) *Data modeling: benefits, types, importance & steps involved*. <https://www.zucisystems.com/blog/what-is-data-modeling-and-why-is-it-important/>.

Team, P.B. (no date b) *What is data modelling?* <https://powerbi.microsoft.com/en-my/what-is-data-modeling/>.

Watt, A. (2014b) *Chapter 7 The Relational Data model*. <https://opentextbc.ca/dbdesign01/chapter/chapter-7-the-relational-data-model/#:~:text=The%20relational%20data%20model%20was%20modern%20commercial%20database%20management%20systems>.

Newcomb, P.H. and Couch, R. (2010b) 'Veterans Health Administration’s VISTA MUMPS modernization pilot,' in *Elsevier eBooks*, pp. 301–345. <https://doi.org/10.1016/b978-0-12-374913-0.00012-3>.

Khan, S. (no date b) *Brief history of the relational model*. <https://www.scribd.com/document/94827565/Brief-History-of-the-Relational-Model>.

Peterson, R. (2024b) *Relational Data model in DBMS | Database Concepts & example*. <https://www.guru99.com/relational-data-model-dbms.html>.

GfG (2024b) *Relational model in DBMS*. <https://www.geeksforgeeks.org/relational-model-in-dbms/>.

T, N. (2021b) *Relational Data model*. <https://binaryterms.com/relational-data-model.html>.