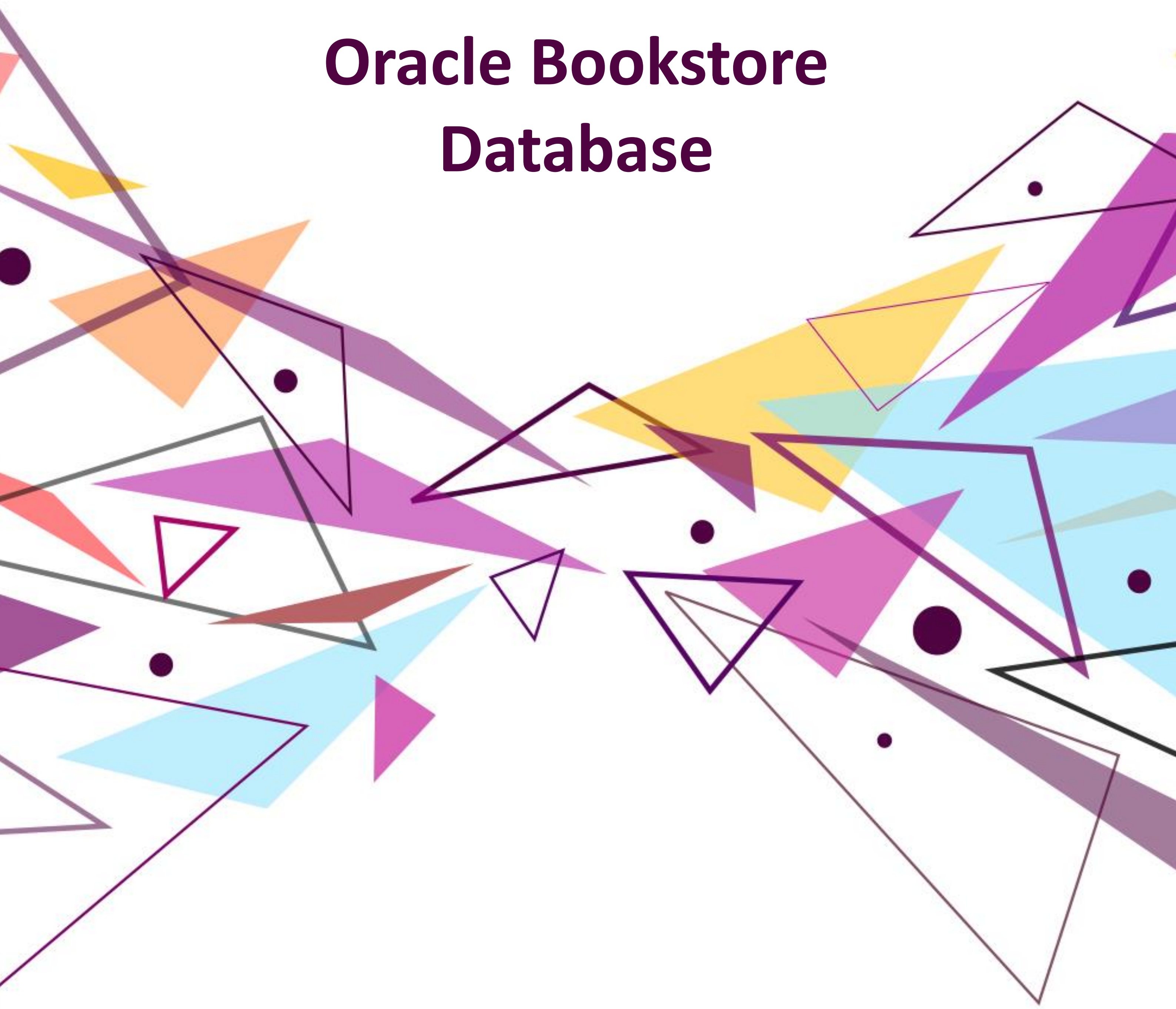
Oracle Bookstore Database



DBA 381: Project

Group:

- Boitumelo Mathabathe (578041)
- Chaleigh Storm (577716)
- Vutivi Maswanganyi (577800)
- Zoë Treutens (577989)

Due: 31/05/2024

Table of Contents

Introduction

In today's data-driven world, the role of an Oracle Database Administrator (DBA) is more critical than ever. The Oracle DBA is the unsung hero who ensures that the database systems, which are the backbone of many organizations, run smoothly and efficiently. This project aims to provide a comprehensive understanding of the various responsibilities and tasks that an Oracle DBA undertakes, from installing and configuring the database to maintaining its performance and security.

An Oracle DBA is tasked with a variety of essential duties, including resolving performance problems, addressing data corruption, and managing hardware failures. These professionals need a deep understanding of database architecture to troubleshoot complex issues swiftly and effectively. Their work ensures that the Oracle Database remains reliable, scalable, and secure, supporting the overall operations of the organization.

This project will take a walkthrough of the practical steps involved in database administration. Starting with the design of an Entity Relationship Diagram (ERD) for an online bookstore. Next, the guide will dive into the hands-on experience of installing and configuring an Oracle Database on a chosen platform, be it Linux or Windows.

The creation and configuration of the database to meet specific organizational needs will also be explored, including setting up tables, views, and indexes. Furthermore, how to populate the database with sample data using the SQL*Loader utility and write effective SQL queries to retrieve meaningful information will be discussed.

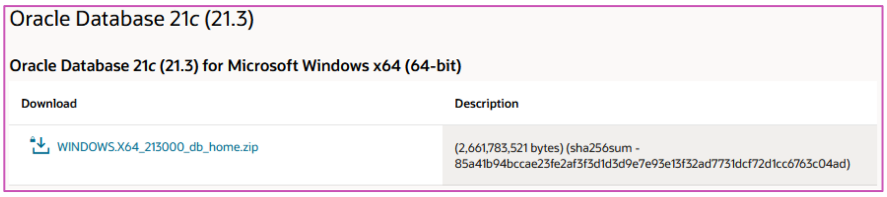
Security is of paramount concern in database administration. This project will guide the user through implementing robust security measures to protect the database from unauthorized access, including creating users and roles, assigning privileges, and enforcing password policies.

Finally, the project emphasizes the importance of monitoring the database's performance and security. Documentation of the process of monitoring the database over a week, identifying any issues that arise and detailing how they were resolved will be created.

By the end of this project, the user will have a solid foundation in Oracle Database Administration, equipped with the knowledge and skills to ensure the smooth operation of an Oracle Database. Whether the user is new to Oracle or looking to refine their DBA skills, this project offers a practical and engaging learning experience.

Oracle Database Installation & Configuration

1. Download the Oracle Database: Follow the link to the Oracle webpage and download the Oracle Database 21c for Microsoft Windows.

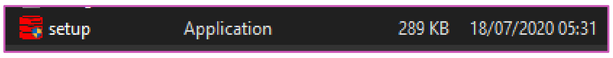


Oracle Database 21c (21.3)

Oracle Database 21c (21.3) for Microsoft Windows x64 (64-bit)

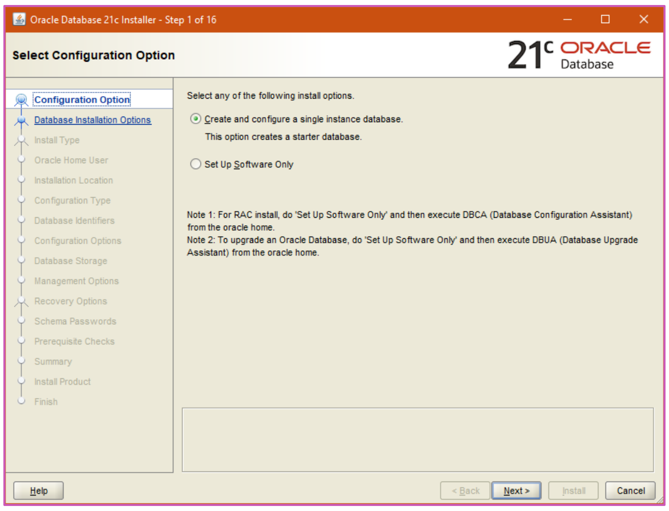
Download	Description
WINDOWS.X64_213000_db_home.zip	(2,661,783,521 bytes) (sha256sum - 85a41b94bccae23fe2af3f3d1d3d9e7e93e13f32ad7731dcf72d1cc6763c04ad)

2. Extract and Run the Installer: Extract the file and run the setup application as administrator to start the installation process.



setup	Application	289 KB	18/07/2020 05:31

3. Choose Installation Type: The oracle installer asks whether to create and configure or install the software only. Due to it being the first-time installing Oracle, choose the "Create and configure a single instance database" and select the Next button.



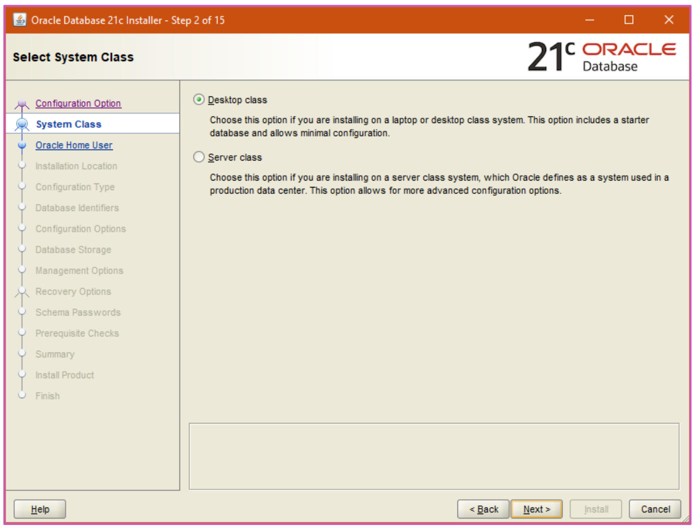
Select any of the following install options.

[✓ Create and configure a single instance database.]
This option creates a starter database.

Set Up Software Only

Note 1: For RAC install, do 'Set Up Software Only' and then execute DBCA (Database Configuration Assistant) from the oracle home.
Note 2: To upgrade an Oracle Database, do 'Set Up Software Only' and then execute DBUA (Database Upgrade Assistant) from the oracle home.

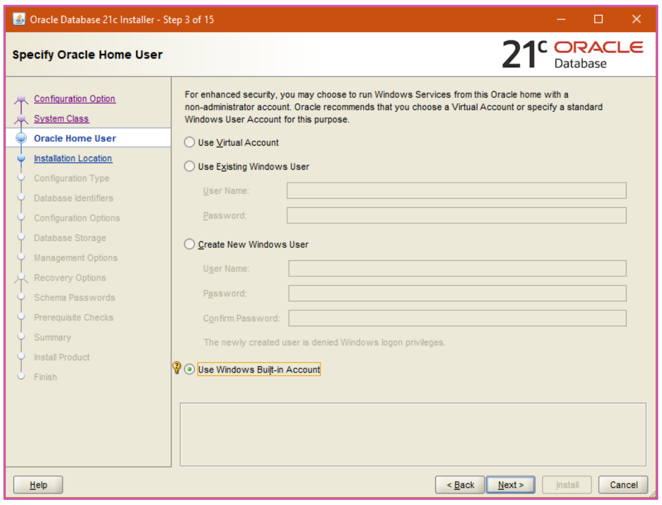
4. Choose System Class: Select the option that installs the oracle database on the computer, "Desktop Class" and select the Next button.



[✓ Desktop class]
Choose this option if you are installing on a laptop or desktop class system. This option includes a starter database and allows minimal configuration.

Server class
Choose this option if you are installing on a server class system, which Oracle defines as a system used in a production data center. This option allows for more advanced configuration options.

5. Specify Windows User Account: In this step, specify whether the user will log in through the built-in account, using an existing account or create an account. In this case select the “Use Windows Built-in Account” option and select the next button.



Oracle Database 21c Installer - Step 3 of 15

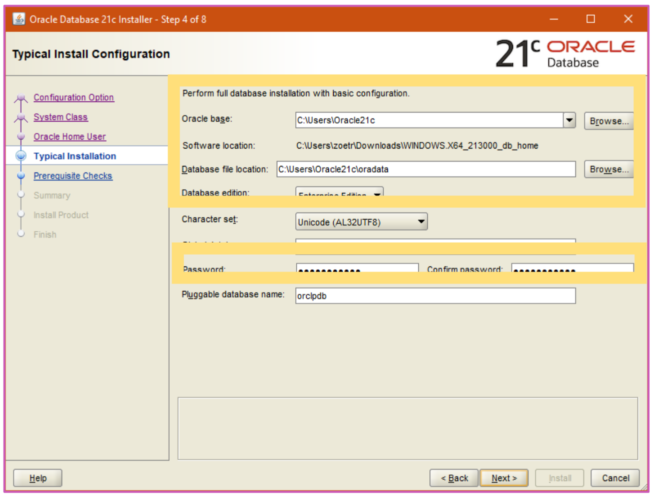
Specify Oracle Home User

21c ORACLE Database

For enhanced security, you may choose to run Windows Services from this Oracle home with a non-administrator account. Oracle recommends that you choose a Virtual Account or specify a standard Windows User Account for this purpose.

- Use Virtual Account
- Use Existing Windows User
  - User Name:
  - Password:
- Create New Windows User
  - User Name:
  - Password:
  - Confirm Password:
  - The newly created user is denied Windows logon privileges.
- [✓ Use Windows Built-in Account]

6. Specify Installation Location and Database Details: In this step, (1) choose the folder where the Oracle database will be stored. (2) Provide a password that will be used to connect to the database. It is recommended that this password be stored in a text file within Oracle21c folder.



Oracle Database 21c Installer - Step 4 of 8

Typical Install Configuration

21c ORACLE Database

Perform full database installation with basic configuration.

Oracle base: C:\Users\Oracle21c

Software location: C:\Users\zoetr\Downloads\WINDOWS.X64_213000_db_home

Database file location: C:\Users\Oracle21c\oradata

Database edition:

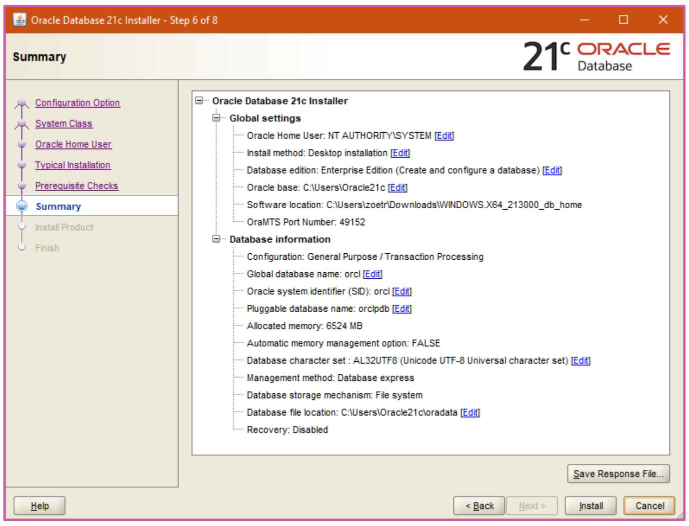
Character set: Unicode (AL32UTF8)

Password: ●●●●●●●●●●●● Confirm password: ●●●●●●●●●●●●

Pluggable database name: orclpdb

7. Prerequisite Checks: The installer will perform prerequisite checks which will some time.

8. Review and Install: This displays the summary of all the information to be installed. Once reviewed select the install button.



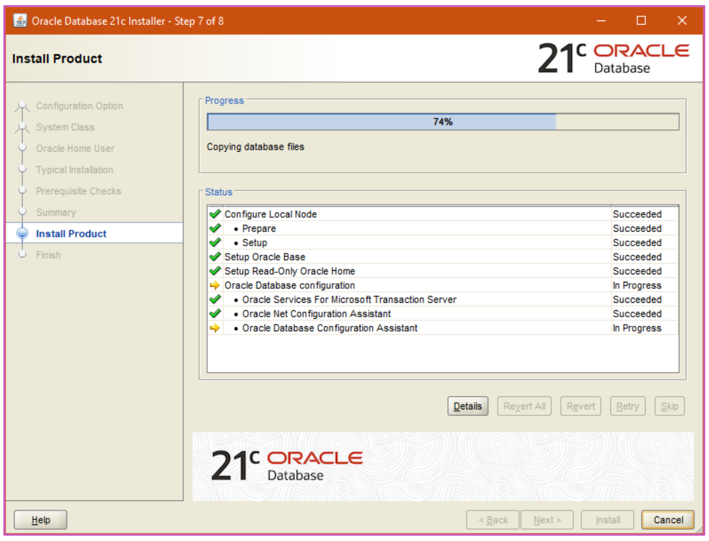
Oracle Database 21c Installer - Step 6 of 8

Summary

21c ORACLE Database

- Oracle Database 21c Installer
  - Global settings
    - Oracle Home User: NT AUTHORITY\SYSTEM [Edit]
    - Install method: Desktop installation [Edit]
    - Database edition: Enterprise Edition (Create and configure a database) [Edit]
    - Oracle base: C:\Users\Oracle21c [Edit]
    - Software location: C:\Users\zoetr\Downloads\WINDOWS.X64_213000_db_home
    - OraMTS Port Number: 49152
  - Database information
    - Configuration: General Purpose / Transaction Processing
    - Global database name: orcl [Edit]
    - Oracle system identifier (SID): orcl [Edit]
    - Pluggable database name: orclpdb [Edit]
    - Allocated memory: 6524 MB
    - Automatic memory management option: FALSE
    - Database character set : AL32UTF8 (Unicode UTF-8 Universal character set) [Edit]
    - Management method: Database express
    - Database storage mechanism: File system
    - Database file location: C:\Users\Oracle21c\oradata [Edit]
    - Recovery: Disabled

9. Installation Process: The installation process will take some time based on the computer.



Progress

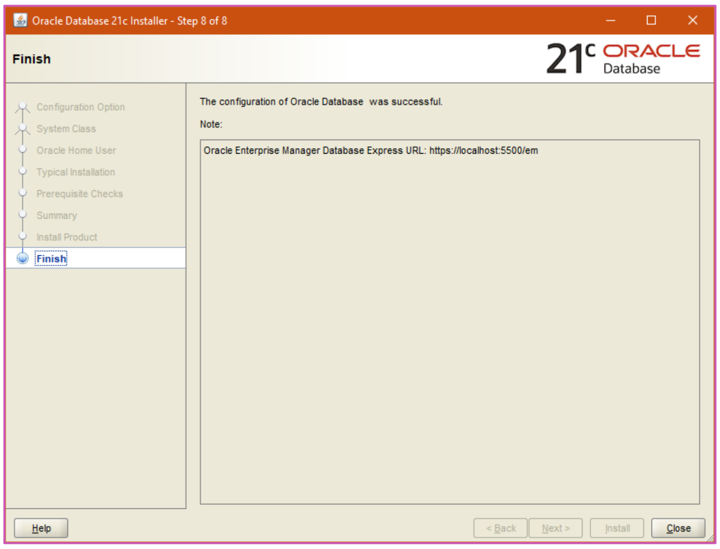
74%

Copying database files

Status

✓ Configure Local Node	Succeeded
✓ • Prepare	Succeeded
✓ • Setup	Succeeded
✓ Setup Oracle Base	Succeeded
✓ Setup Read-Only Oracle Home	Succeeded
⇨ Oracle Database configuration	In Progress
✓ • Oracle Services For Microsoft Transaction Server	Succeeded
✓ • Oracle Net Configuration Assistant	Succeeded
⇨ • Oracle Database Configuration Assistant	In Progress

10. Installation Completion: The database is finished installing.

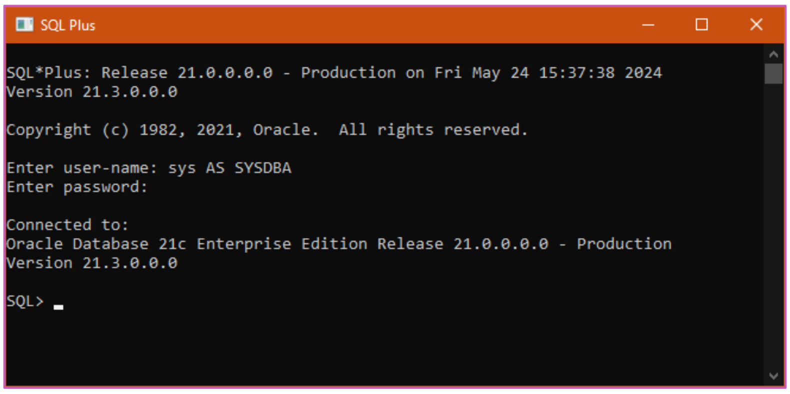


The configuration of Oracle Database was successful.

Note:

Oracle Enterprise Manager Database Express URL: https://localhost:5500/em

11. Connect to Oracle Database: In the search bar of the computer type “SQL Plus” and open the command prompt. In the command prompt it will ask for the username and password created when the database was installed. As the built-in account was selected use “sys AS SYSDBA” for the username and the password that was entered for the password. Connecting to the database was successful.



```
SQL*Plus: Release 21.0.0.0.0 - Production on Fri May 24 15:37:38 2024
Version 21.3.0.0.0

Copyright (c) 1982, 2021, Oracle.  All rights reserved.

Enter user-name: sys AS SYSDBA
Enter password:

Connected to:
Oracle Database 21c Enterprise Edition Release 21.0.0.0.0 - Production
Version 21.3.0.0.0

SQL> _
```

Following these steps will ensure a successful installation and configuration of the Oracle database on a Windows system.

Database Design

Database Requirements

To implement a database for the online bookstore, Thabo decided to use an Oracle database due to its robustness, scalability, and security features. Tables with established relationship using foreign keys and views containing frequently accessed data, such as the most popular books and top customers are created. Indexes can be created on frequently searched columns to speed up query performance.

Thabo's Online Bookstore has five main entities:

1. Book: This table stored information about each book, including its title, their ISBN number, who the author and publisher are, the date it was published, the price of the book, and the quantity in stock.
2. Author: This table stored information about each author, including their name and ID.
3. Order: This table stored information about each customer order, including the date and time of the order and the customer's ID.
4. Order Detail: This table stored information about each item ordered, including the book ISBN number, quantity ordered, and the price at the time of the order.
5. Customer: This table stored information about each customer, including their name, address, and email.

The relationships between these entities are as follows:

- A book could have ***one or more*** authors, and an author could write ***one or more*** books.
- An order could have ***one or more*** order details, and an order detail belonged to ***exactly one*** order.
- A book could include ***zero or more*** order details, and an order detail includes ***exactly one*** book.
- A customer could place ***zero or more*** orders, and an order belonged to ***exactly one*** customer.

Security: Thabo can create two user roles: ***Customer*** and ***Admin***. Customers have the privileges to browse books, place orders, and update their personal information. Admins, on the other hand, will have the privilege to add or remove books, manage orders, and access customer information. Database-level access control policies should be put into place in addition to the user roles. Sensitive data, such as private client information, should be encrypted and access to it should be closely monitored.

As for the Database itself the following requirements can be deduced and utilised when creating an effective and efficient database for the bookstore.

Files and Filegroups: Several files and filegroups should be created within the database to improve management and performance. Multiple files can be stored in each filegroup. For example, the Order, Order Detail, and Customer tables can potentially be placed in a separate filegroup (secondary), while the frequently visited Book and Author tables can be placed in another filegroup (primary).

File Size and Growth: The database files should have an initial size that allows for the anticipated volume of data. For example, the initial size of the file containing the Book table should be large enough to store information for approximately 10,000 books. When the available space is about to run out, the file size will automatically rise if the file growth is set to auto-growing mode. A reasonable growth increment of 10% should be established to avoid numerous auto-growth operations, which might negatively impact performance.

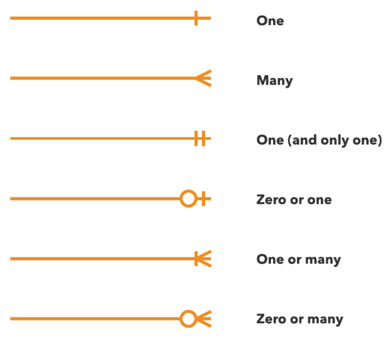
Log Files: Transaction log files are essential for disaster recovery and maintaining the integrity of the database. The number of transactions affects the log file's size. A large log file would be essential for an online shop that processes a lot of transactions, such as orders for books. Like the database files, the log file must also be in auto-growth mode.

Backup and Recovery: It is important to plan regular backups to avoid data loss. To reduce the impact on performance, differential backups can be carried out during working hours and full backups during off-peak hours. For safety, the backup files must be kept on an independent hard drive.

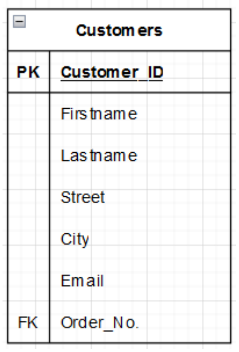
Modelling: Entity Relationship Diagram (ERD)

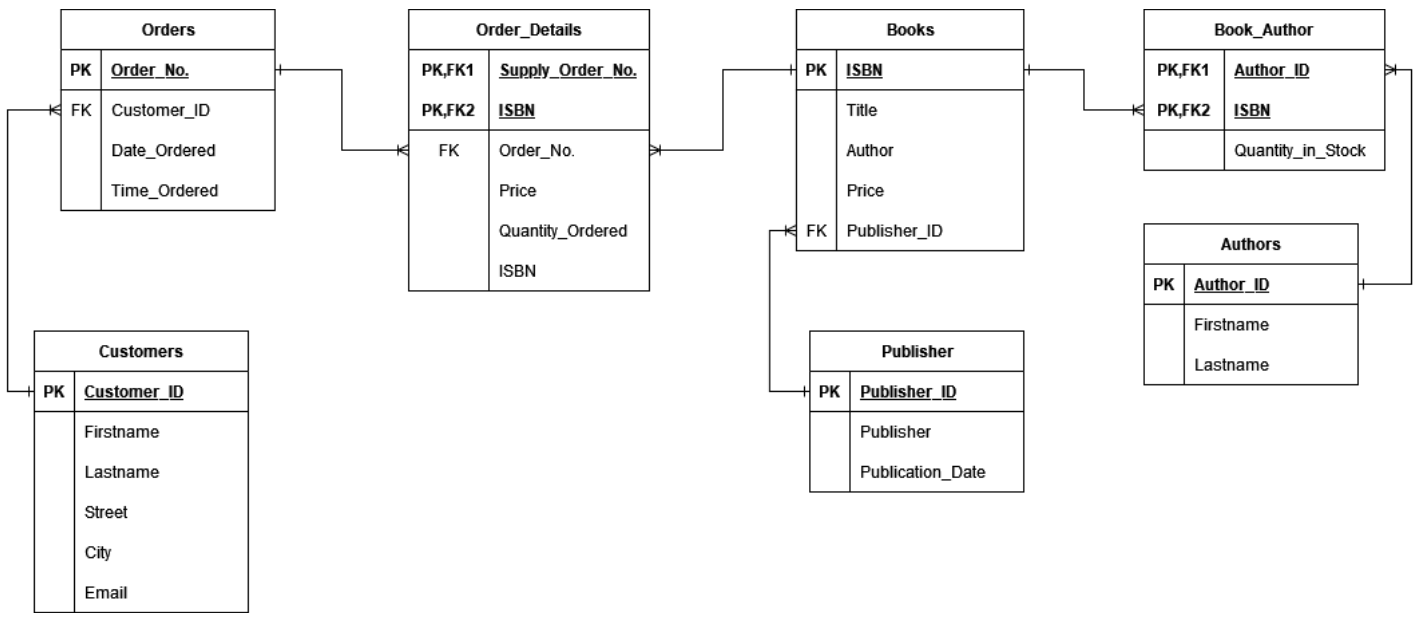
An entity relationship diagram is a visual representation of a databases entity structures and the relationships between them.

ERD Cardinality Notations:



Entity:





Database Creation & Configuration

- In the Oracle database, a new tablespace is defined using the CREATE TABLESPACE command. The tablespace will act as a logical storage structure to contain the various tables, indexes, and views to be created. The tablespace to be created is the “BookStore” tablespace.
- The storage location for the “BookStore” is specified using the clause DATAFILE and followed by the path 'C:\Users\Oracle21c\oradata\BookStore_GroupE\bookstore_groupE.dbf' that points to the specific directory on the disk.

- The datafiles initial size will be set to 10M (megabytes). When the file reaches its maximum capacity, the AUTOEXTEND ON clause will enable it to automatically grow by a specified increment in the tablespace. This in turn will prevent any issues related to inserting or updating data in the file when the tablespace reaches its maximum capacity.
- The "BookStore" tablespace's expected data volume influenced the decision to start with an initial size of 10MB. It is customary to begin with a smaller size at first and allow for automatic expansion as required later without any intervention from the DBA.

```
Group_E > ALTER SESSION SET CONTAINER = xepdb1;
Session altered.

Group_E > CREATE TABLESPACE BookStore
  2  DATAFILE 'C:\Users\zoetr\OracleXE\oradata\BookStore_GroupE\bookstore_groupE.dbf'
  3  SIZE 10M AUTOEXTEND ON;
Tablespace created.
```

Database Content

I. Customers Table

- **Attributes:**
  - Customer_ID: A unique identifier for each customer (e.g., 16931).
  - Firstname: The first name of the customer (e.g., Hannah).
  - Lastname: The last name of the customer (e.g., Mitchell).
  - Street: The street address of the customer (e.g., 3606 Robinson Loaf).
  - City: The city where the customer resides (e.g., Swansonberg).
  - Email: The email address of the customer (e.g., allenjames@example.org).

- **Attribute Datatypes:**
  - Customer_ID: NUMBER
  - Firstname: VARCHAR2(100)
  - Lastname: VARCHAR2(100)
  - Street: VARCHAR2(200)
  - City: VARCHAR2(200)
  - Email: VARCHAR2(100)

- **Dependencies:** A customer can place ***zero or more*** orders.
- **Keys:** The customer table has a single primary key: Customer_ID
- **Index:**
  - On the attributes "Firstname" and "Lastname," an index is created called idx_customer_name to speed up any searches made based on the customer's name.

```
Group_E > CREATE INDEX idx_customer_name ON Customers (Firstname, Lastname);

Index created.
```

- **Schema:**

```
Group_E > DESCRIBE Customers;

 Name                 Null?          Type
 -------------------- -------------  ----------------------
 CUSTOMER_ID          NOT NULL       NUMBER
 FIRSTNAME            NOT NULL       VARCHAR2(100)
 LASTNAME             NOT NULL       VARCHAR2(100)
 STREET               NOT NULL       VARCHAR2(200)
 CITY                 NOT NULL       VARCHAR2(200)
 EMAIL                NOT NULL       VARCHAR2(100)
```

- **Creation Code:**

```
Group_E > CREATE TABLE Customers (
  2   Customer_ID NUMBER PRIMARY KEY,
  3   Firstname VARCHAR2(100) NOT NULL,
  4   Lastname VARCHAR2(100) NOT NULL,
  5   Street VARCHAR2(200) NOT NULL,
  6   City VARCHAR2(200) NOT NULL,
  7   Email VARCHAR2(100) NOT NULL
  8  );

Table created.
```

- **Data Display:**
    - 50 sample data records were inserted into the table, but the first 10 records are shown in the table below.

```
Group_E > COLUMN FIRSTNAME FORMAT A10 WORD_WRAPPED
Group_E > COLUMN LASTNAME FORMAT A10 WORD_WRAPPED
Group_E > COLUMN STREET FORMAT A15 WORD_WRAPPED
Group_E > COLUMN CITY FORMAT A15 WORD_WRAPPED
Group_E > COLUMN EMAIL FORMAT A25 WORD_WRAPPED
Group_E > SELECT * FROM Customers FETCH FIRST 10 ROWS ONLY;
```

Customer_ID	Firstname	Lastname	Street	City	Email
16931	Hannah	Mitchell	3606 Robinson Loaf	Swansonberg	allenjames@example.org
28605	Melanie	Mckay	6098 Huynh Highway Apt. 245	East Adam	cfernandez@example.com
30643	Justin	Salazar	27010 Moon Ford Apt. 012	New Michelle	hayesjennifer@example.org
97782	Michael	Jackson	57236 John Via	New Anashire	nicholsonkevin@example.org
37693	Edward	Jordan	775 Taylor Road	Port Jennifer	ahart@example.org
44488	Rhonda	Brandt	38990 Martinez Knoll	Karenville	laurenward@example.com
86695	Lisa	Barr	4646 Julia Cliffs	West Christopher	pamtodd@example.com
92844	Samantha	Manning	34807 Brian Mountains Apt. 300	Garciafort	quinnerik@example.net
52933	Caroline	Stone	769 Spencer Square Apt. 243	Katrinaside	qmiller@example.com

10 rows selected.

II. Author Table

- **Attributes:**
    - Author_ID: A unique identifier for various authors (e.g., 60699).
    - Firstname: The first name of the author (e.g., Michelle).
    - Lastname: The last name of the author (e.g., Thomas).

- **Attribute Datatypes:**
    - Author_ID: NUMBER
    - Firstname: VARCHAR2 (100)

  - Lastname: VARCHAR2(100)

- **Dependencies:** An author can write one or more books.
- **Keys:** The author table has a primary Key Author_ID.
- **Index:**
  - A combined index on the “Firstname” and “Lastname” attributes was created called idx_author_name to optimize searches based on author names.

```
Group_E > CREATE INDEX idx_author_name ON Author (Firstname, Lastname);
Index created.
```

- **Schema:**

```
Group_E > DESCRIBE Author;

Name               Null?       Type
------------------ --------    ----------------------------
AUTHOR_ID          NOT NULL    NUMBER
FIRSTNAME          NOT NULL    VARCHAR2(100)
LASTNAME           NOT NULL    VARCHAR2(100)
```

- **Creation Code:**

```
Group_E > CREATE TABLE Author (
  2   Author_ID NUMBER PRIMARY KEY,
  3   Firstname VARCHAR2(100) NOT NULL,
  4   Lastname VARCHAR2(100) NOT NULL
  5  );

Table created.
```

- **Data Display:**
  - 50 sample data records were inserted into the table, but the first 10 records are shown in the table below.

```
Group_E > COLUMN FIRSTNAME FORMAT A10 WORD_WRAPPED
Group_E > COLUMN LASTNAME FORMAT A15 WORD_WRAPPED
Group_E > SELECT * FROM Author FETCH FIRST 10 ROWS ONLY;
```

Author_ID	Firstname	Lastname
60699	Michelle	Thomas
81596	Sabrina	Lee
62733	Joseph	Pennington
29568	Kevin	Morales
47943	William	Nelson
40198	Joshua	Smith
7426	Breanna	Lloyd
43991	Jeremy	Rivera
3957	Brian	Stephenson

10 rows selected.

III. Book_Author Table

- **Attributes:**
  - Author_ID: The identifier of the author (e.g., 60699).
  - ISBN: The book's unique identifier (e.g., 79748).
  - Quantity_in_Stock: The quantity of the book in stock (e.g., 85).

- **Attribute Datatypes:**
  - Author_ID: NUMBER
  - ISBN: VARCHAR2(20)
  - Quantity_in_Stock: NUMBER

- **Dependencies:**
  - The Author_ID attribute has a foreign key constraint FK_AuthorID that references the Author table's Author_ID.
  - The ISBN attribute has a foreign key constraint FK_ISBN that references the Book table's ISBN.

- **Keys:** A composite primary key was created with the Author_ID and ISBN attributes.

```
Group_E > ALTER TABLE Book_Author ADD CONSTRAINT FK_AuthorID FOREIGN KEY (Author_ID) REFERENCES
Author(Author_ID);
Table altered.

Group_E > ALTER TABLE Book_Author ADD CONSTRAINT FK_ISBN FOREIGN KEY (ISBN) REFERENCES Book(ISBN);
Table altered.
```

- **Schema:**

```
Group_E > DESCRIBE Book_Author;

 Name                          Null?          Type
 ---------------------------   --------------  --------------------
 AUTHOR_ID                     NOT NULL       NUMBER
 ISBN                          NOT NULL       VARCHAR2(20)
 QUANTITY_IN_STOCK             NOT NULL       NUMBER
```

- **Creation Code:**

```
Group_E > CREATE TABLE Book_Author (
  Author_ID NUMBER,
  ISBN VARCHAR2(20),
  Quantity_in_Stock NUMBER NOT NULL,
  PRIMARY KEY (Author_ID, ISBN)
 );

Table created.
```

- **Data Display:**
  - 50 sample data records were inserted into the table, but the first 10 records are shown in the table below.

```
Group_E > SELECT * FROM Book_Author FETCH FIRST 10 ROWS ONLY;
```

Author_ID	ISBN	Quantity_in_Stock
60699	79748	85
81596	87449	37
37837	48682	99
85949	82240	74
32	49089	41
33	88557	46
69	63125	4
43991	26259	81
3957	34238	75
51429	97738	93

10 rows selected.

IV. Book Table

- **Attributes:**
  - ISBN: A unique identifier for books (e.g., 34238).
  - Title: The title of the book (e.g., Quality-focused 5thgeneration strategy).
  - Author_ID: The identifier of the book's author (e.g., 51).
  - Price: The price of the book (e.g., 26).
  - Publisher_ID: The identifier of the book's publisher (e.g., 19698).

- **Attribute Datatypes:**
  - ISBN: VARCHAR2 (20)
  - Title: VARCHAR2 (100)
  - Author_ID: NUMBER
  - Price: NUMBER
  - Publisher_ID: VARCHAR2 (20)

- **Dependencies:**
  - The Publisher_ID attribute has a foreign key constraint FK_PublisherID that references the Publisher table's Publisher_ID.
  - A book can be included in ***zero or more*** order details.

- **Keys:**
  - The primary key is the ISBN attribute.
  - The foreign key in the table is the Publisher_ID

```
Group_E > ALTER TABLE Book ADD CONSTRAINT FK_PublisherID FOREIGN KEY (Publisher_ID) REFERENCES
Publisher(Publisher_ID);

Table altered.
```

- **Index:**
  - An index named idx_book_title was created on the Title attribute to speed up searches based on the book titles.

```
Group_E > CREATE INDEX idx_book_title ON Book (Title);

Index created.
```

- **Schema:**

```
Group_E > DESCRIBE Book;

 Name                  Null?          Type
 --------------------- -------------  -----------------------
 ISBN                  NOT NULL       VARCHAR2(20)
 TITLE                 NOT NULL       VARCHAR2(100)
 AUTHOR_ID             NOT NULL       NUMBER
 PRICE                                NUMBER
 PUBLISHER_ID                         VARCHAR2(20)
```

- **Creation Code:**

```
Group_E > CREATE TABLE Book (
  ISBN VARCHAR2(20) PRIMARY KEY,
  Title VARCHAR2(100) NOT NULL,
  Author_ID NUMBER NOT NULL,
  Price NUMBER,
  Publisher_ID VARCHAR2(20)
 );

Table created.
```

- **Data Display:**
  - 50 sample data records were inserted into the table, but the first 10 records are shown in the table below.

```
Group_E > SELECT * FROM Book_Author FETCH FIRST 10 ROWS ONLY;
```

Author_ID	ISBN	Quantity_in_Stock
60699	79748	85
81596	87449	37
37837	48682	99
85949	82240	74
32	49089	41
33	88557	46
69	63125	4
43991	26259	81
3957	34238	75
51429	97738	93

10 rows selected.

V. Publisher Table

- **Attributes:**
  - Publisher_ID: A unique identifier for publishers (e.g., 31814).
  - Publisher: The name of the publisher (e.g., Watson, Bender, and Erickson).
  - Publish_Date: The date of publication (e.g., 01/11/2019).

- **Attribute Datatypes:**
  - Publisher_ID: VARCHAR2(20)
  - Publisher: VARCHAR2(100)
  - Publish_Date: DATE

- **Dependencies:** A publisher can publish ***zero or more*** books.

- **Keys:** The primary key in the Publisher table is called Publisher_ID.
- **Schema Screenshot:**

```
Group_E > DESCRIBE Publisher;

 Name                  Null?         Type
 --------------------- ------------- ----------------------
 PUBLISHER_ID          NOT NULL      VARCHAR2(20)
 PUBLISHER             NOT NULL      VARCHAR2(100)
 PUBLISH_DATE                        DATE
```

- **Creation Code:**

```
Group_E > CREATE TABLE Publisher(
  2   Publisher_ID VARCHAR2(20) PRIMARY KEY,
  3   Publisher VARCHAR2(100) NOT NULL,
  4   Publish_Date DATE
  5  );

Table created.
```

- **Data Display:**
  - 50 sample data records were inserted into the table, but the first 10 records are shown in the table below.

```
Group_E > SELECT * FROM Publisher FETCH FIRST 10 ROWS ONLY;
```

Publisher_ID	Publisher	Publish_Date
19698	Brewer-Stone	22/09/1996
10005	Young, Smith and Moreno	16/07/2018
55689	Patterson and Sons	02/05/2004
7501	Price-Munoz	15/11/2004
86689	Wolf-Brock	21/02/2019
9018	Moreno-White	03/08/2013
78929	Gibbs LLC	22/03/2015
64894	Davis-Woodard	22/08/2000
29130	Medina, King and Brown	15/05/2006
18759	Clark-Brown	20/08/1997

```
10 rows selected.
```

VI. Orders Table

- **Attributes:**
  - Order_ID: A unique identifier for orders (e.g., 22307).
  - Order_Date: The date when the order was placed (e.g., 17/12/2023).
  - Customer_ID: The identifier of the customer placing the order (e.g., 97782).

- **Attribute Datatypes:**
  - Order_ID: NUMBER
  - Order_Date: DATE
  - Customer_ID: NUMBER

- **Dependencies:**
  - The Customer_ID attribute has a foreign key constraint FK_CustomerID that references the Customers table's Customer_ID.

- An order can be in ***one or more*** order details.
- An order belonged to ***exactly one*** customer.

- **Keys:**
  - The primary key in the Orders table is the Order_ID and the foreign key is the Customer_ID.

```
Group_E > ALTER TABLE Orders ADD CONSTRAINT FK_CustomerID FOREIGN KEY (Customer_ID) REFERENCES
Customers(Customer_ID);

Table altered.
```

- **Schema Screenshot:**

```
Group_E > DESCRIBE Orders;

 Name                 Null?          Type
 -------------------- -------------- -----------------
 ORDER_ID             NOT NULL       NUMBER
 ORDER_DATE           NOT NULL       DATE
 CUSTOMER_ID          NOT NULL       NUMBER
```

- **Creation Code:**

```
Group_E > CREATE TABLE Orders (
  2   Order_ID NUMBER PRIMARY KEY,
  3   Order_Date DATE NOT NULL,
  4   Customer_ID NUMBER NOT NULL
  5  );

Table created.
```

- **Data Display:**
  - 50 sample data records were inserted into the table, but the first 10 records are shown in the table below.

```
Group_E > SELECT * FROM Orders FETCH FIRST 10 ROWS ONLY;
```

Order_ID	Order_Date	Customer_ID
22307	17/12/2023	97782
35407	21/11/2023	37693
15147	10/07/2023	44488
84750	31/07/2023	86695
1554	20/07/2023	92844
52421	20/11/2023	52933
95090	03/09/2023	20994
45284	15/12/2023	39371
1174	03/07/2023	2794
24540	11/09/2023	41783

10 rows selected.

VII. Order Details Table

- **Attributes:**
  - Supply_Order_ID: The identifier of the supply order (e.g., 71543).
  - ISBN: The book's unique identifier (e.g., 34782).

    - Order_ID: The identifier of the order (e.g., 84750).
    - Quantity: The quantity of the book ordered (e.g., 72).
    - Price: The price of the book in the order (e.g., 72).

- **Attribute Datatypes:**
    - Supply_Order_ID: NUMBER
    - ISBN: VARCHAR2(20)
    - Order_ID: NUMBER
    - Quantity: NUMBER
    - Price: NUMBER

- **Dependencies:**
    - The Order_ID attribute has a foreign key constraint FK_OrderID that references the Orders table's Order_ID.
    - The ISBN attribute has a foreign key constraint FK_ISBN_OrderDetail that references the Book table's ISBN.
    - An order detail includes ***exactly one*** book.
    - An order detail belonged to ***exactly one*** order.

- **Keys:**
    - Composite Primary Key: (Supply_Order_ID, ISBN)

```
Group_E > ALTER TABLE OrderDetail ADD CONSTRAINT FK_OrderID FOREIGN KEY (Order_ID) REFERENCES
Orders(Order_ID);
Table altered.

Group_E > ALTER TABLE OrderDetail ADD CONSTRAINT FK_ISBN_OrderDetail FOREIGN KEY (ISBN) REFERENCES
Book(ISBN);
Table altered.
```

- **Schema:**

```
Group_E > DESCRIBE OrderDetail;

 Name                      Null?         Type
 ------------------------- ------------- ---------------------
 SUPPLY_ORDER_ID           NOT NULL      NUMBER
 ISBN                      NOT NULL      VARCHAR2(20)
 ORDER_ID                  NOT NULL      NUMBER
 QUANTITY                  NOT NULL      NUMBER
 PRICE                                   NUMBER
```

- **Creation Code:**

```
Group_E > CREATE TABLE OrderDetail (
  Supply_Order_ID NUMBER,
  ISBN VARCHAR2(20),
  Order_ID NUMBER NOT NULL,
  Quantity NUMBER NOT NULL,
  Price NUMBER,
  PRIMARY KEY (Supply_Order_ID, ISBN)
  );

Table created.
```

- **Data Display:**
    - 50 sample data records were inserted into the table, but the first 10 records are shown in the table below.

```
Group_E > SELECT * FROM OrderDetail FETCH FIRST 10 ROWS ONLY;
```

Supply_Order_ID	ISBN	Order_ID	Quantity	Price
71543	34782	84750	72	72
68673	92012	1554	23	26
31284	63125	52421	20	91
91738	48682	95090	93	27
28510	82240	45284	17	21
94897	47531	1174	21	64
16751	54296	24540	23	26
8998	91987	11349	80	63
1057	58392	78711	20	13
37382	97738	12502	12	16

10 rows selected.

VIII. Most Popular Books View

- **Schema:**

```
Group_E > DESCRIBE MOSTPOPULARBOOKS;
 Name                      Null?          Type
 ------------------------- -------------- ----------------------------
 TITLE                     NOT NULL       VARCHAR2(100)
 NUMBEROFORDERS                           NUMBER
```

- **Creation Code:**

```
Group_E > CREATE VIEW MostPopularBooks AS
  2  SELECT Book.Title, COUNT(OrderDetail.Order_ID) AS NumberOfOrders
  3  FROM Book JOIN OrderDetail ON Book.ISBN = OrderDetail.ISBN
  4  GROUP BY Book.Title
  5  ORDER BY NumberOfOrders DESC;

Table created.
```

- **Data Display:**

```
Group_E > SQL> SET PAGESIZE 1000
Group_E > COLUMN TITLE FORMAT A60 WORD_WRAPPED
Group_E > SELECT * FROM MostPopularBooks;
```

Title	Number of Orders
Multi-channelled well-modulated Graphical User Interface	3
Persevering non-volatile conglomeration	2
Pre-emptive bifurcated projection	2
Inverse user-facing framework	2
Realigned global system engine	2
Multi-lateral 3rdgeneration customer loyalty	2
Devolved hybrid strategy	2
Diverse asymmetric matrices	1
Self-enabling zero administration firmware	1
Self-enabling fault-tolerant instruction set	1
Configurable asynchronous circuit	1
Implemented systemic structure	1
Cross-platform transitional capacity	1

Monitored even-keeled concept	1
Operative motivating service-desk	1
Virtual zero administration encryption	1
Persistent disintermediate alliance	1
Integrated object-oriented challenge	1
Networked user-facing structure	1
Operative bi-directional productivity	1
Horizontal systematic benchmark	1
Enhanced systematic architecture	1
Integrated systematic solution	1
Devolved even-keeled pricing structure	1
Multi-tiered empowering forecast	1
Profound interactive circuit	1
Object-based secondary system engine	1
Exclusive 24hour algorithm	1
User-centric motivating info-mediaries	1
Reactive multi-state utilization	1
Synergistic full-range website	1
Profound heuristic flexibility	1
Quality-focused 5thgeneration strategy	1
Customer-focused tertiary application	1

34 rows selected.

IX. Top Customers View

- **Schema:**

```
Group_E > DESCRIBE TopCustomers;

 Name                          Null?              Type
 ----------------------------  -----------------  -------------------------------
 FIRSTNAME                      NOT NULL           VARCHAR2(100)
 LASTNAME                       NOT NULL           VARCHAR2(100)
 NUMBEROFORDERS                                    NUMBER
```

- **Creation Code:**

```
Group_E > CREATE VIEW TopCustomers AS
  2  SELECT Customers.Firstname, Customers.Lastname, COUNT(Orders.Order_ID) AS NumberOfOrders
  3  FROM Customers JOIN Orders ON Customers.Customer_ID = Orders.Customer_ID
  4  GROUP BY Customers.Firstname, Customers.Lastname
  5  ORDER BY NumberOfOrders DESC;

Table created.
```

- **Data Display:**
  - 50 sample data records were inserted into the table, but the first 10 records are shown in the table below.

```
Group_E > SET PAGESIZE 1000
Group_E > COLUMN FIRSTNAME FORMAT A10 WORD_WRAPPED
Group_E > COLUMN LASTNAME FORMAT A20 WORD_WRAPPED
Group_E > SELECT * FROM TopCustomers;
```

Firstname	Lastname	Number of Orders
Rhonda	Brandt	2
Heather	Sexton	2
Lisa	Miller	2
Joshua	Gilbert	2
Don	Franklin	2
Lisa	Barr	2
Edward	Jordan	2
Michael	Jackson	2
William	Hernandez	2
Caroline	Stone	2
Samantha	Manning	2
Victoria	Morrison	2
Russell	Daniels	1
James	Cochran	1
Monica	Friedman	1
Timothy	Anderson	1
Sherry	Hernandez	1
Thomas	Miller	1
Oscar	Lewis	1
Matthew	Alvarado	1
Debra	Williams	1
Jonathan	Nichols	1
Keith	Valentine	1
David	Sanchez	1
Terri	Moreno	1
Karen	Harding	1
Alec	Christensen	1
Stacie	Hale	1
Nathaniel	Chapman	1
Kimberly	Thomas	1
Emily	Hodge	1
Kimberly	Frazier	1
Phillip	Mccarty	1
Justin	Salazar	1
Jennifer	Graves	1
Hannah	Mitchell	1
Melanie	Mckay	1

37 rows selected.

Data Population and Retrieval

Database Population Process

Having created and looked at the structure and schema of the various tables and views, the process of data population can now take place. The process of data population involves inserting data into tables within a database.

50 records of sample data for each table were generated using python scrips:

Libraries:

```
import csv
from faker import Faker
import random
from datetime import datetime, timedelta

fake = Faker()
```

Book Table

```
with open('CSV Files/book.csv', 'w', newline='') as file:
  writer = csv.writer(file)
  writer.writerow(["ISBN", "Title", "Author_ID", "Price", "Publisher_ID"])
  for _ in range(50):
    writer.writerow([fake.unique.random_number(digits=5), fake.catch_phrase(), fake.random_int(min=1, max=100),
fake.random_number(digits=2, fix_len=True), fake.random_number(digits=5)])
```

Publisher Table

```
with open('CSV Files/publisher.csv', 'w', newline='') as file:
  writer = csv.writer(file)
  writer.writerow(["Publisher_ID", "Publisher", "Publish_Date"])
  for _ in range(50):
    writer.writerow([fake.unique.random_number(digits=5), fake.company(), fake.date_between(start_date='-30y', end_date='today')])
```

Author Table

```
with open('CSV Files/author.csv', 'w', newline='') as file:
  writer = csv.writer(file)
  writer.writerow(["Author_ID", "Firstname", "Lastname"])
  for _ in range(50):
    writer.writerow([fake.unique.random_number(digits=5), fake.first_name(), fake.last_name()])
```

Book_Author Table

```
with open('CSV Files/book_author.csv', 'w', newline='') as file:
  writer = csv.writer(file)
  writer.writerow(["Author_ID", "ISBN", "Quantity_in_Stock"])
  for _ in range(50):
    writer.writerow([fake.random_int(min=1, max=100), fake.random_number(digits=5), fake.random_int(min=1, max=100)])
```

Order Table

```
with open('CSV Files/order.csv', 'w', newline='') as file:
  writer = csv.writer(file)
  writer.writerow(["Order_ID", "Order_Date", "Customer_ID"])
  for _ in range(50):
    writer.writerow([fake.unique.random_number(digits=5), fake.date_between(start_date='-1y', end_date='today'),
fake.random_int(min=1, max=100)])
```

OrderDetails Table

```
with open('CSV Files/order_detail.csv', 'w', newline='') as file:
  writer = csv.writer(file)
  writer.writerow(["Supply_Order_ID", "ISBN", "Order_ID", "Quantity", "Price"])
```

```
    for _ in range(50):
        writer.writerow([fake.unique.random_number(digits=5), fake.random_number(digits=5), fake.random_int(min=1, max=100),
fake.random_int(min=1, max=100), fake.random_number(digits=2, fix_len=True)])
```

Customers Table

```
with open('CSV Files/customers.csv', 'w', newline='') as file:
    writer = csv.writer(file)
    writer.writerow(["Customer_ID", "Firstname", "Lastname", "Street", "City", "Email"])
    for _ in range(50):
        writer.writerow([fake.unique.random_number(digits=5), fake.first_name(), fake.last_name(), fake.street_address(), fake.city(),
fake.email()])
```

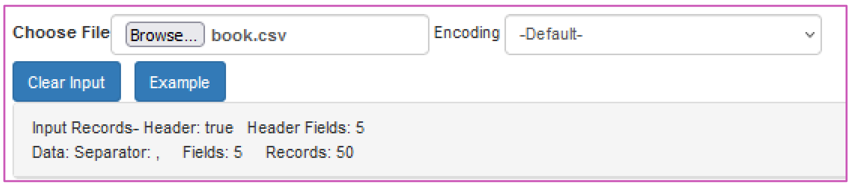
The generated data are saved into csv files.

The generated data can be uploaded into the various tables in the following ways:

1. SQL*Loader
    - Oracle has a powerful tool called SQL*Loader, or sqlldr, which makes it simple to load huge amounts of data from external files into the Oracle Database tables.

    - How to use SQL*Loader:
        - **Control File:** This file is used to specify the data loading procedure. The instructions in this text file instructs the SQL*Loader on how to comprehend the data file.
        - **Data File:** This is the external file containing the sample data to be loaded into the tables.
        - **Command Line:** Start the tool by using the sqlldr command line: sqlldr control=my_control_file.ctl data=my_data_file.csv
        - **Data Conversion:** Data fields in the data file are converted into database columns.
        - **Logging:** SQL*Loader records details regarding the load process in a log file.

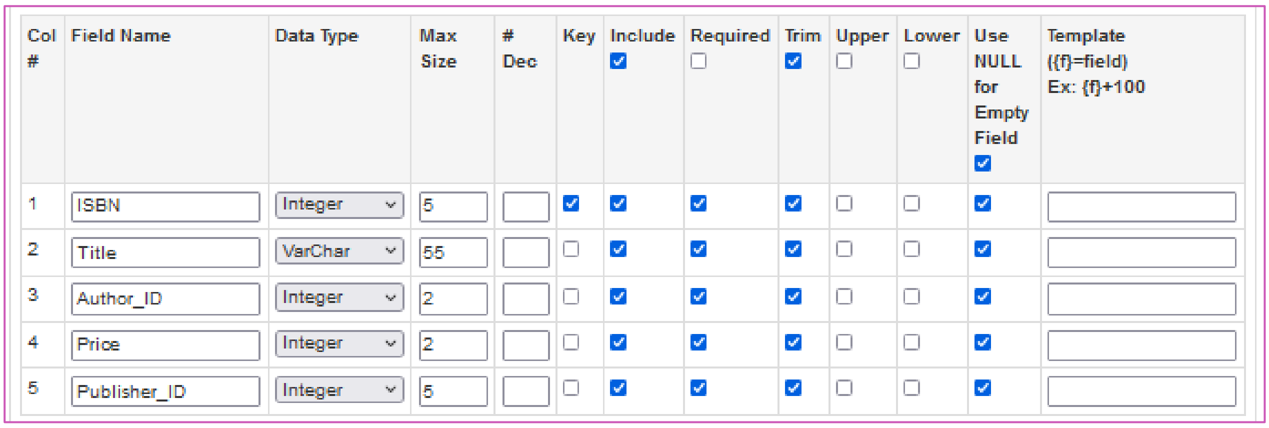
2. INSERT Statements
    - Using the website Convert CSV to SQL created by Data Design Group, the csv files are converted into SQL INSERT statements.
    - Step 1: Upload the csv file:



Choose File: book.csv — Encoding: -Default-

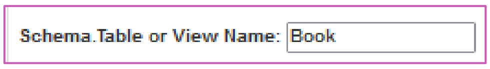
Input Records- Header: true   Header Fields: 5
Data: Separator: ,   Fields: 5   Records: 50

    - Step 2: Confirm the attributes have the correct datatypes and field sizing:



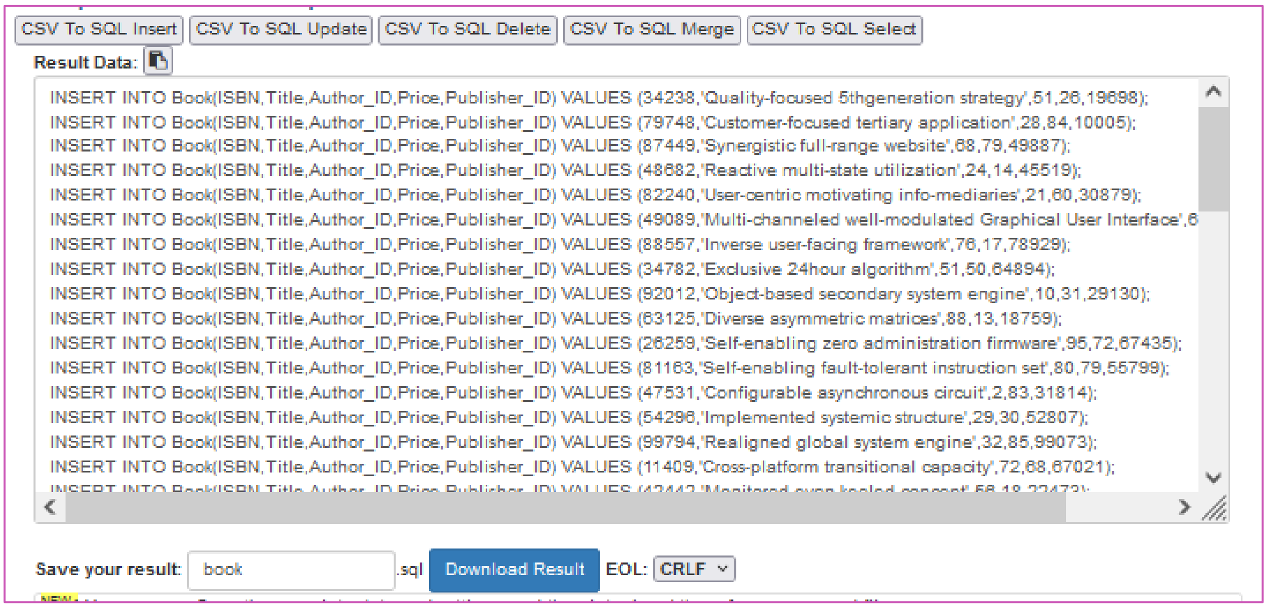
Col #	Field Name	Data Type	Max Size	# Dec	Key	Include ✓	Required ☐	Trim ✓	Upper ☐	Lower ☐	Use NULL for Empty Field ✓	Template ({f}=field) Ex: {f}+100
1	ISBN	Integer	5		✓	✓	✓	✓	☐	☐	✓	
2	Title	VarChar	55		☐	✓	✓	✓	☐	☐	✓	
3	Author_ID	Integer	2		☐	✓	✓	✓	☐	☐	✓	
4	Price	Integer	2		☐	✓	✓	✓	☐	☐	✓	
5	Publisher_ID	Integer	5		☐	✓	✓	✓	☐	☐	✓	

- Step 3: Provide the table with a name:



Schema.Table or View Name: Book

- Step 4: Select the CSV to SQL Insert option to generate the statements. Save the statements to a CSV file.



```
INSERT INTO Book(ISBN,Title,Author_ID,Price,Publisher_ID) VALUES (34238,'Quality-focused 5thgeneration strategy',51,26,19698);
INSERT INTO Book(ISBN,Title,Author_ID,Price,Publisher_ID) VALUES (79748,'Customer-focused tertiary application',28,84,10005);
INSERT INTO Book(ISBN,Title,Author_ID,Price,Publisher_ID) VALUES (87449,'Synergistic full-range website',68,79,49887);
INSERT INTO Book(ISBN,Title,Author_ID,Price,Publisher_ID) VALUES (48682,'Reactive multi-state utilization',24,14,45519);
INSERT INTO Book(ISBN,Title,Author_ID,Price,Publisher_ID) VALUES (82240,'User-centric motivating info-mediaries',21,60,30879);
INSERT INTO Book(ISBN,Title,Author_ID,Price,Publisher_ID) VALUES (49089,'Multi-channeled well-modulated Graphical User Interface',6
INSERT INTO Book(ISBN,Title,Author_ID,Price,Publisher_ID) VALUES (88557,'Inverse user-facing framework',76,17,78929);
INSERT INTO Book(ISBN,Title,Author_ID,Price,Publisher_ID) VALUES (34782,'Exclusive 24hour algorithm',51,50,64894);
INSERT INTO Book(ISBN,Title,Author_ID,Price,Publisher_ID) VALUES (92012,'Object-based secondary system engine',10,31,29130);
INSERT INTO Book(ISBN,Title,Author_ID,Price,Publisher_ID) VALUES (63125,'Diverse asymmetric matrices',88,13,18759);
INSERT INTO Book(ISBN,Title,Author_ID,Price,Publisher_ID) VALUES (26259,'Self-enabling zero administration firmware',95,72,67435);
INSERT INTO Book(ISBN,Title,Author_ID,Price,Publisher_ID) VALUES (81163,'Self-enabling fault-tolerant instruction set',80,79,55799);
INSERT INTO Book(ISBN,Title,Author_ID,Price,Publisher_ID) VALUES (47531,'Configurable asynchronous circuit',2,83,31814);
INSERT INTO Book(ISBN,Title,Author_ID,Price,Publisher_ID) VALUES (54296,'Implemented systemic structure',29,30,52807);
INSERT INTO Book(ISBN,Title,Author_ID,Price,Publisher_ID) VALUES (99794,'Realigned global system engine',32,85,99073);
INSERT INTO Book(ISBN,Title,Author_ID,Price,Publisher_ID) VALUES (11409,'Cross-platform transitional capacity',72,68,67021);
```

Save your result: book .sql EOL: CRLF

- Step 4: Populate the database tables from the SQL*Plus command prompt:
    - Due to the constraints on the tables, data is inserted from the following order: Customers, Author, Publisher, Book, Book_Author, Orders, OrderDetails

1. Customers Table Insertion: Involves inserting customer information like Customer_ID, Firstname, Lastname, Street, City, and Email into the Customers table.

   Example: **INSERT INTO Customers(Customer_ID, Firstname, Lastname, Street, City, Email) VALUES (16931, 'Hannah', 'Mitchell', '3606 Robinson Loaf', 'Swansonberg', 'allenjames@example.org');**

2. Author Table Insertion: Inserts author information such as Author_ID, Firstname, and Lastname into the Author table.
   Example: **INSERT INTO Author(Author_ID, Firstname, Lastname) VALUES (60699, 'Michelle', 'Thomas');**

3. Publisher Table Insertion: Populates the Publisher table with Publisher_ID, Publisher name, and Publish_Date.
   Example: **INSERT INTO Publisher (Publisher_ID, Publisher, Publish_Date) VALUES (19698, 'Brewer-Stone', TO_DATE('1996-09-22', 'YYYY-MM-DD'));**

4. Book Table Insertion: Inserts book details like ISBN, Title, Author_ID, Price, and Publisher_ID into the Book table.
   Example: **INSERT INTO Book(ISBN, Title, Author_ID, Price, Publisher_ID) VALUES (34238, 'Quality-focused 5thgeneration strategy', 51, 26, 19698);**

5. Book_Author Table Insertion: Establishes the relationship between books and authors, including Author_ID, ISBN, and Quantity_in_Stock.
   Example: **INSERT INTO Book_Author(Author_ID, ISBN, Quantity_in_Stock) VALUES (60699, 79748, 85);**

6. Orders Table Insertion: Inserts order details like Order_ID, Order_Date, and Customer_ID into the Orders table.
   Example: **INSERT INTO Orders (Order_ID, Order_Date, Customer_ID) VALUES (22307, TO_DATE('2023-12-17', 'YYYY-MM-DD'), '97782')**

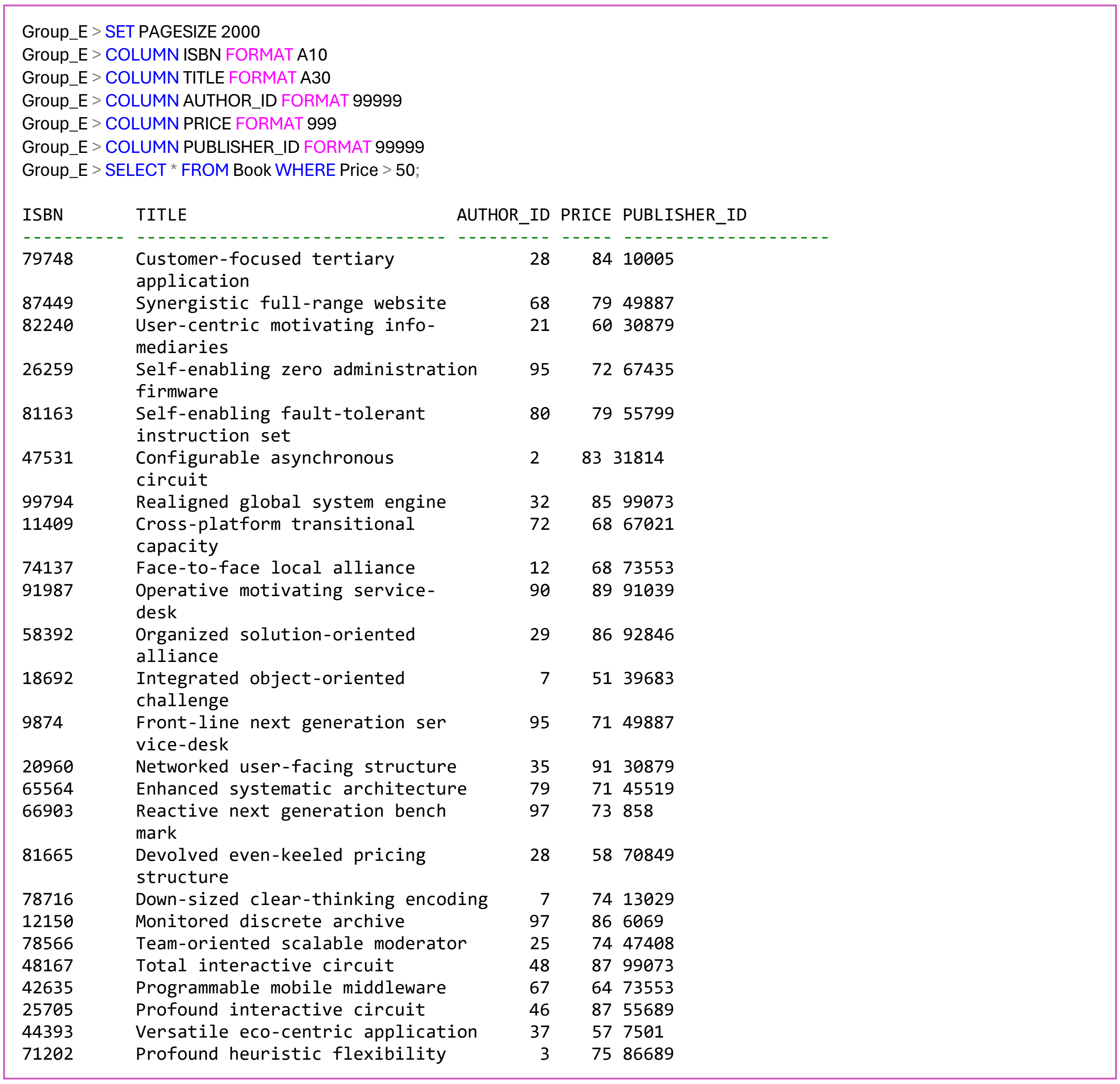
7. OrderDetail Table Insertion: Records information about ordered items including Supply_Order_ID, ISBN, Order_ID, Quantity, and Price.
   Example: **INSERT INTO OrderDetail(Supply_Order_ID, ISBN, Order_ID, Quantity, Price) VALUES (71543, 34782, 84750, 72, 72);**

Database Queries

With all the tables created and the sample data inserted into the various tables, queries can now be created to view the data in many ways to draw insights into the information and use it to make data driven decisions.

I. Select Queries

- Display all the books that are more expensive than R50.
  - This query will allow Thabo to analyse the books that are priced the highest and could consider offering discounts or promotions to boost sales.



```
Group_E > SET PAGESIZE 2000
Group_E > COLUMN ISBN FORMAT A10
Group_E > COLUMN TITLE FORMAT A30
Group_E > COLUMN AUTHOR_ID FORMAT 99999
Group_E > COLUMN PRICE FORMAT 999
Group_E > COLUMN PUBLISHER_ID FORMAT 99999
Group_E > SELECT * FROM Book WHERE Price > 50;

ISBN       TITLE                          AUTHOR_ID PRICE PUBLISHER_ID
---------- ------------------------------ --------- ----- --------------------
79748      Customer-focused tertiary             28    84 10005
           application
87449      Synergistic full-range website        68    79 49887
82240      User-centric motivating info-         21    60 30879
           mediaries
26259      Self-enabling zero administration     95    72 67435
           firmware
81163      Self-enabling fault-tolerant          80    79 55799
           instruction set
47531      Configurable asynchronous            2    83 31814
           circuit
99794      Realigned global system engine        32    85 99073
11409      Cross-platform transitional           72    68 67021
           capacity
74137      Face-to-face local alliance           12    68 73553
91987      Operative motivating service-         90    89 91039
           desk
58392      Organized solution-oriented           29    86 92846
           alliance
18692      Integrated object-oriented             7    51 39683
           challenge
9874       Front-line next generation ser        95    71 49887
           vice-desk
20960      Networked user-facing structure       35    91 30879
65564      Enhanced systematic architecture      79    71 45519
66903      Reactive next generation bench        97    73 858
           mark
81665      Devolved even-keeled pricing          28    58 70849
           structure
78716      Down-sized clear-thinking encoding     7    74 13029
12150      Monitored discrete archive            97    86 6069
78566      Team-oriented scalable moderator      25    74 47408
48167      Total interactive circuit             48    87 99073
42635      Programmable mobile middleware        67    64 73553
25705      Profound interactive circuit          46    87 55689
44393      Versatile eco-centric application     37    57 7501
71202      Profound heuristic flexibility         3    75 86689
```

```
1808        Customizable asynchronous system      28     96 49641
            engine
79046       Open-architected contextually-        80     95 17527
            based service-desk

27 rows selected.
```

- Display all the customers that live in the cities starting with an ‘N.’
  - Thabo can gain insight into which cities have the highest concentration of customers.

```
Group_E > SET PAGESIZE 7000
Group_E > COLUMN CUSTOMER_ID FORMAT 99999
Group_E > COLUMN FIRSTNAME FORMAT A10
Group_E > COLUMN LASTNAME FORMAT A10
Group_E > COLUMN STREET FORMAT A20
Group_E > COLUMN CITY FORMAT A15
Group_E > COLUMN EMAIL FORMAT A30
Group_E > SELECT * FROM Customers WHERE City LIKE 'N%' ;

CUSTOMER_ID FIRSTNAME  LASTNAME   STREET               CITY
----------- ---------- ---------- -------------------- ---------------
EMAIL
------------------------------
      30643 Justin     Salazar    27010 Moon Ford Apt. New Michelle
                                   012
hayesjennifer@example.org
      97782 Michael    Jackson    57236 John Via       New Anashire
nicholsonkevin@example.org
      88752 Robert     Richardson 36184 Anthony Island North Virginia
michael89@example.net
      44779 Timothy    Anderson   53981 Paula Oval     New Mitchell
                                  Suite 882
michaelcooper@example.com
      13930 Sherry     Hernandez  525 Dawn Parkway     New Richardfurt
waguilar@example.net
      41997 Duane      Coleman    274 Adams Knolls     New Edward
                                  Suite 674
bthompson@example.net
      19979 Keith      Valentine  169 Williams        New David
                                  Turnpike Suite 020
meganewing@example.net

7 rows selected.
```

II. Delete Queries

- A delete query can be used to delete a specific customer from the Orders table using their Order_ID
- Additionally, orders can be deleted from the OrderDetails table using the Order_ID’s.
- Orders with a price less than R20 are deleted from the OrderDetails table.

```
Group_E > DELETE FROM Orders WHERE Order_ID = 100;
0 rows deleted.

Group_E > DELETE FROM OrderDetail WHERE Order_ID = 190;
1 row deleted.

Group_E > DELETE FROM OrderDetail WHERE Price < 20;
6 rows deleted.
```

III. Insert Query

- A new order’s details can be inserted into the OrderDetails table.

```
Group_E > INSERT INTO OrderDetail (Supply_Order_ID, ISBN, Order_ID, Quantity, Price) VALUES (48129, 31781, 13877, 62,
13);

1 row created.
```

IV. Update Query

- The price of a book can be updated using the ISBN number.
  - This will provide Thabo with the ability to keep his records up to date.

```
Group_E > UPDATE Book SET Price = 30 WHERE ISBN = 42035;

1 row updated.
```

V. Join Queries

- Display the book title and the aggregate quantity of all the books sold.
  - Thabo can analyse the sales to determine which books have the highest sales.
  - He can also track the demand of the highest selling books to ensure they are adequately stocked in store.

```
Group_E > SELECT Book.Title, SUM(OrderDetail.Quantity) AS TotalQuantitySold
FROM Book
JOIN OrderDetail ON Book.ISBN = OrderDetail.ISBN
GROUP BY Book.Title;

TITLE                                              TOTALQUANTITYSOLD
-------------------------------------------------- -----------------
Quality-focused 5thgeneration strategy                            89
Customer-focused tertiary application                             91
Synergistic full-range website                                    12
Reactive multi-state utilization                                  93
User-centric motivating info-mediaries                            17
Multi-channeled well-modulated Graphical User Interface          154
Inverse user-facing framework                                    107
Exclusive 24hour algorithm                                        72
Object-based secondary system engine                              23
Diverse asymmetric matrices                                       20
Self-enabling zero administration firmware                        60
Self-enabling fault-tolerant instruction set                      37
Configurable asynchronous circuit                                 21
Implemented systemic structure                                    23
Realigned global system engine                                    98
Cross-platform transitional capacity                              16
Monitored even-keeled concept                                     94
Multi-lateral 3rdgeneration customer loyalty                     103
Operative motivating service-desk                                 80
Virtual zero administration encryption                            23
Persistent disintermediate alliance                               23
Integrated object-oriented challenge                              52
Networked user-facing structure                                   99
Operative bi-directional productivity                             31
Pre-emptive bifurcated projection                                129
Persevering non-volatile conglomeration                          158
Devolved hybrid strategy                                          36
Horizontal systematic benchmark                                    7
Enhanced systematic architecture                                  39
Integrated systematic solution                                    48
Devolved even-keeled pricing structure                            54
Multi-tiered empowering forecast                                  33
Profound interactive circuit                                      54
Profound heuristic flexibility                                    41

34 rows selected.
```

- Determine the total amount spent by each customer.
  - Thabo can identify loyal customers based on their amount spent to offer rewards or discounts to them.
  - He can also use this to target the higher spending customers with premium book offers.

```
Group_E > SELECT Customers.Firstname, Customers.Lastname, SUM(OrderDetail.Quantity * Book.Price) AS TotalSpent
FROM Customers
JOIN Orders ON Customers.Customer_ID = Orders.Customer_ID
JOIN OrderDetail ON Orders.Order_ID = OrderDetail.Order_ID
JOIN Book ON OrderDetail.ISBN = Book.ISBN
GROUP BY Customers.Firstname, Customers.Lastname;

FIRSTNAME  LASTNAME             TOTALSPENT
---------- -------------------- ----------
Keith      Valentine                  2314
Stacie     Hale                      14764
Victoria   Morrison                    948
William    Hernandez                  2815
Emily      Hodge                      1020
Alec       Christensen                1220
Monica     Friedman                   1720
Lisa       Miller                     2909
Caroline   Stone                      2870
Lisa       Barr                       3600
Samantha   Manning                     713
Melanie    Mckay                      4320
Nathaniel  Chapman                    3613
Kimberly   Thomas                     1743
James      Cochran                     765
Joshua     Gilbert                    9983
Phillip    Mccarty                    3392
Kimberly   Frazier                    6390
Terri      Moreno                     1178
David      Sanchez                     779
Heather    Sexton                      782
Russell    Daniels                     644
Justin     Salazar                    2652
Don        Franklin                   9629
Thomas     Miller                     1960
Edward     Jordan                     1156
Sherry     Hernandez                    68
Jennifer   Graves                      266
Hannah     Mitchell                   3132
Jonathan   Nichols                    1056
Oscar      Lewis                      3075

31 rows selected.
```

- This query will display the title of the book and the author's full name.
  - This will assist customers who are searching for a book based on the author's name and visa versa.
  - It can also assist Thabo with order fulfilment and tracking of publisher details.

```
Group_E > COLUMN TITLE FORMAT A45
Group_E > COLUMN PUBLISHER FORMAT A30
Group_E > SELECT Book.Title, Publisher.Publisher FROM Book JOIN Publisher ON Book.Publisher_ID = Publisher.Publisher_ID;

TITLE                                         PUBLISHER
--------------------------------------------- ------------------------------
Quality-focused 5thgeneration strategy        Brewer-Stone
Customer-focused tertiary application         Young, Smith and Moreno
Pre-emptive bifurcated projection             Patterson and Sons
Profound interactive circuit                  Patterson and Sons
Persevering non-volatile conglomeration       Price-Munoz
Versatile eco-centric application             Price-Munoz
Profound heuristic flexibility                Wolf-Brock
Multi-channeled well-modulated Graphical User Moreno-White
Interface
Inverse user-facing framework                 Gibbs LLC
Exclusive 24hour algorithm                    Davis-Woodard
Object-based secondary system engine          Medina, King and Brown
Diverse asymmetric matrices                   Clark-Brown
Self-enabling zero administration firmware    Hanson LLC
Self-enabling fault-tolerant instruction set  Fuller Group
Configurable asynchronous circuit             Watson, Bender and Erickson
Implemented systemic structure                Taylor Inc
Integrated object-oriented challenge          Sharp Group
Devolved hybrid strategy                      Sharp Group
Synergistic full-range website                Leblanc Group
Front-line next generation service-desk       Leblanc Group
```

```
Horizontal systematic benchmark               Leblanc Group
Reactive multi-state utilization              Flores-Turner
Digitized fresh-thinking methodology          Flores-Turner
Enhanced systematic architecture              Flores-Turner
Operative motivating service-desk             Silva, Bryant and Patterson
Organized solution-oriented alliance          Martin, Charles and Floyd
Virtual zero administration encryption        King-Hoffman
Persistent disintermediate alliance           Bowen LLC
User-centric motivating info-mediaries        Lowe PLC
Networked user-facing structure               Lowe PLC
Operative bi-directional productivity         Deleon, Davis and Huynh
Reactive next generation benchmark            Welch LLC
Integrated systematic solution                Buchanan, Schmidt and Martin
Devolved even-keeled pricing structure        Rodriguez, Coleman and Jones
Down-sized clear-thinking encoding            Pitts Ltd
Monitored discrete archive                    Hardin LLC
Team-oriented scalable moderator              Molina LLC
Fundamental grid-enabled Internet solution    Powell, Young and Jackson
Cross-platform logistical website             Solis-Burke
Realigned global system engine                Wright, Anderson and Schwartz
Total interactive circuit                     Wright, Anderson and Schwartz
Cross-platform transitional capacity          Adams-Shelton
Multi-tiered empowering forecast              Adams-Shelton
Monitored even-keeled concept                 Patel, Hendrix and Miller
Profit-focused high-level extranet            Patel, Hendrix and Miller
Face-to-face local alliance                   Hunter and Sons
Programmable mobile middleware                Hunter and Sons
Multi-lateral 3rdgeneration customer loyalty  Schmidt-Bradford
Customizable asynchronous system engine       Morales-Carter
Open-architected contextually-based service   White LLC
-desk

50 rows selected.
```

Data Security Measures

There are three users created for this database tablespace "BookStore":

1. Thabo
   - Using the systems login details and logging into the database as the systems administrator (SYSDBA), the user Thabo is created.
   - Thabo was given a password: A1B2C3D4 and was made the database administrator for the tablespace "BookStore".
   - Thabo was granted the following control:
     - **DBA:** This privilege will provide Thabo with extensive management activities in the database.
     - **Connect:** Allows him to connect to the database.
     - **Resource:** Provides the ability to create and manage schema objects.
     - **Alter Session:** Allows Thabo to modify the sessions settings.
     - **Create tablespace:** This will allow him to create a new tablespace.

```
Group_E > CREATE USER Thabo IDENTIFIED BY A1B2C3D4 DEFAULT TABLESPACE BookStore;
User created.

Group_E > GRANT DBA TO Thabo;
Grant succeeded.

Group_E > GRANT CONNECT, RESOURCE TO Thabo;
Grant succeeded.

Group_E > GRANT ALTER SESSION TO Thabo;
Grant succeeded.

Group_E > GRANT CREATE TABLESPACE TO Thabo;
Grant succeeded.
```

2. Bob
    - The user Bob was created with the password 1234.
    - Bob is assigned to the tablespace BookStore.

```
Group_E > CREATE USER Bob IDENTIFIED BY 1234 DEFAULT TABLESPACE BookStore;
User created.
```

3. Martha
    - The user Martha was created with the password ABCD.
    - Martha is assigned to the tablespace BookStore.

```
Group_E > CREATE USER Martha IDENTIFIED BY ABCD DEFAULT TABLESPACE BookStore;
User created.
```

There are 2 roles created for this database tablespace “BookStore”:

1. Customers
    - The role “Customers” will be created for the customers of the online bookstore to access the database.
    - “Customers” have the privilege to browse books, place orders, and update their personal information.
    - This role is assigned to the user Bob.

```
Group_E > CREATE ROLE Customer;
Role created.

Group_E > GRANT SELECT ON Book TO Customer;
Grant succeeded.

Group_E > GRANT SELECT, UPDATE ON Customers TO Customer;
Grant succeeded.

Group_E > GRANT SELECT, INSERT ON Orders TO Customer;
Grant succeeded.

Group_E > GRANT SELECT, INSERT ON OrderDetail TO Customer;
Grant succeeded.

Group_E > GRANT Customer TO Bob;
Grant succeeded.
```

2. Admin
    - The role of “Admins” will be created for the staff members of the online bookstore to access the database.
    - “Admins” will have the privilege to add or remove books, manage orders, and access customer information.
    - This role is assigned to the user Martha.

```
Group_E > CREATE ROLE Admin;
Role created.

Group_E > GRANT SELECT, INSERT, UPDATE, DELETE ON Book TO Admin;
Grant succeeded.

Group_E > GRANT SELECT, INSERT, UPDATE, DELETE ON Publisher TO Admin;
Grant succeeded.

Group_E > GRANT SELECT, INSERT, UPDATE, DELETE ON Author TO Admin;
Grant succeeded.
```

```
Group_E > GRANT SELECT, INSERT, UPDATE, DELETE ON Book_Author TO Admin;
Grant succeeded.

Group_E > GRANT SELECT, INSERT, UPDATE, DELETE ON OrderDetail TO Admin;
Grant succeeded.

Group_E > GRANT SELECT ON Customers TO Admin;
Grant succeeded.

Group_E > GRANT SELECT, INSERT, UPDATE, DELETE ON Orders TO Admin;
Grant succeeded.

Group_E > GRANT Admin TO Martha;
Grant succeeded.
```

Password Policies

Database-level access control policies should be put into place in addition to the users and roles created. Sensitive data, such as private client information, should be encrypted and access to it should be closely monitored.

The users Thabo, Bob, and Martha are assigned to a security profile that was created for the tablespace. This profile will set limits that the users must abide by to access and perform various operations on the database. The security profile checks are as follows:

1. **Failed login attempts:** The maximum number of times a user can log in unsuccessfully is three times before they are logged out of the database for 24hours. This will prevent forceful hacking attempts into the database by an unauthorised user.

2. **Lifetime of the password:** The password has a 30-day duration before the user must change the password. This will enhance the security of the database by reducing the risk of credentials being compromised.
3. **Reusability of the password:** Once a password is changed, the user can only reuse the password 1 more time after 180 days. This will encourage the creation of stronger passwords.

```
Group_E > CREATE PROFILE secure_profile LIMIT
  2   FAILED_LOGIN_ATTEMPTS 3
  3   PASSWORD_LIFE_TIME 30
  4   PASSWORD_REUSE_TIME 180
  5   PASSWORD_REUSE_MAX 1;
Profile created.

Group_E > ALTER USER Thabo PROFILE secure_profile;
User altered.

Group_E > ALTER USER Bob PROFILE secure_profile;
User altered.

Group_E > ALTER USER Martha PROFILE secure_profile;
User altered.

Group_E > ALTER SYSTEM SET RESOURCE_LIMIT = TRUE;
System altered.
```

Overall, the strict regulation of passwords through locking the user out if they unsuccessfully logged in; the frequent change of passwords for each user; and limiting the reusability of the password to prevent overusing and security leaks will add to the security and privacy features of the database.

Database Monitoring and Issue Resolution

Monitoring Activities

I. Tablespace Usage

- This task involves checking the storage consumption and the free space available in the BookStore tablespace.
- This will benefit Thabo as he can proactively monitor the storage resources to ensure it is not taking up excessive amounts of space or requires more space.
- This will optimise the performance of the tablespace.
- Thabo can also set up an alert to notify him if the tablespace will run out of storage before it happens.

```
SELECT tablespace_name, used_space, free_space
FROM dba_tablespace_usage_metrics
WHERE tablespace_name = 'BOOKSTORE';
```

II. Recent Activity

- Reviews the recent activity made through insert, update, and delete statements in the tables of the tablespace.
- This will benefit the Bookstore database's owner as he can track spikes in activities that might lead to overworking and performance degradation.

- Unauthorised changes can be tracked and prevented to ensure the databases security is secure.

```
Group_E > SELECT table_name, inserts, updates, deletes
  FROM dba_tab_modifications
  WHERE tablespace_name = 'BOOKSTORE';
```

III. Review Alert Logs

- Regularly checks and monitors the databases alert logs for any error messages or warnings to ensure the database is in good health and not experiencing any performance issues.
- Alert logs include important details concerning warnings, database faults, and problems with performance which Thabo can attend to immediately.
- Preventing problems from becoming worse guarantees the database's continued integrity.

```
Group_E > SHOW PARAMETER background_dump_dest
```

IV. Backup & Recovery

- Maintaining a regular backup schedule is essential to prevent data loss and ensures data recovery capabilities.
- Data Preservation: In the event of hardware malfunctions, accidental deletions, or unexpected events, backups guarantee that Thabo's data remains secure.

- Point-in-Time Recovery: Thabo may restore the database back to a specific point in time, such as right before an error occurred.
- Reduced Downtime: Thabo can swiftly recover from backups in the event of data loss, reducing the database's downtime,

```
Group_E > ALTER DATABASE MOUNT;
Group_E > BACKUP TABLESPACE BookStore;
Group_E > ALTER DATABASE OPEN;

Group_E >  ALTER DATABASE MOUNT;
Group_E > RECOVER TABLESPACE BookStore;
Group_E > ALTER DATABASE OPEN;
```

Through these monitoring initiatives, Thabo can keep the online bookshop database encrypted, efficient, and in excellent condition. Through proactive resource management, identification of anomalies, and rapid issue resolution, Thabo can reduce downtime and offer his customers a consistent experience.

APPENDIX

SQL Script	Record Generator	Database Datafile
DBA381_GroupsE_Project_Script.sql	Record Generator.pdf	**BOOKSTORE_GROUPE.DBF**