RIGA TECHNICAL UNIVERSITY
FACULTY OF COMPUTER SCIENCE AND INFORMATION TECHNOLOGY
INSTITUTE OF APPLIED COMPUTER SYSTEMS

Practical Assignment #5
“Database Management Systems”

Advanced SQL Constructs

Author: Emir Oğuz
Course, Group: DSP201, Group 1
Student Card No: 230ADB011

Checked: Andrejs Gaidukovs

2022 / 2023 Study Year

Content

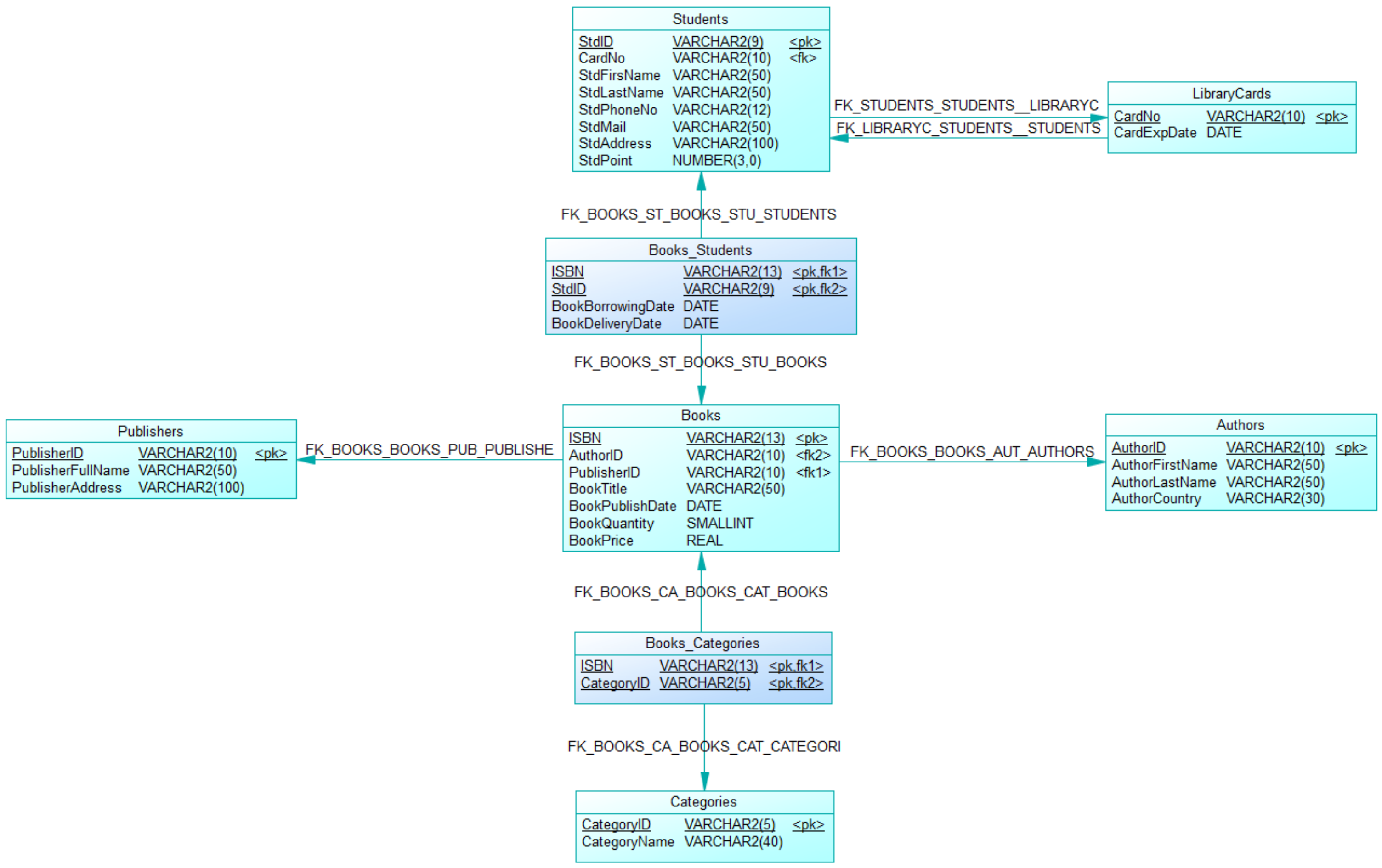
1 Goal

Learn about and use various SQL constructs.

2 Task

1. Create at least 2 views
2. Create at least 2 materialized views
3. Create the following SQL queries:
   1) Using CASE expression – at least 2 queries
   2) Hierarchical queries using:
      - START WITH, CONNECT BY and LEVEL – at least 1 query;
      - WITH, UNION ALL – at least 1 query;
   3) Using analytical functions OVER, PARTITION BY, ORDER BY, RANK, DENSE_RANK, ROW_NUMBER, FIRST, LAST, NTILE, WIDTH_BUCKET – at least 2 queries;
   4) Using windowing function ROWS, RANGE, BETWEEN, UNBOUNDED, PRECEDING, FOLLOWING, CURRENT ROW – at least 2 queries;
   5) Create at least 3 dimensions tables and one fact table and create at least 2 queries using GROUP BY CUBE, GROUPING(), GROUPING SETS;
   6) Create one table with at least 3 dimensions attributes and one fact attribute. Create queries using MODEL, PARTITION BY, DIMENSION BY, MEASURES, RULES – at least 2 queries.
4. Conclusions
5. Submit to Ortus:
   a. Report file in MS Word format names: DBMS_5_Surname.docx
   b. Sql script: DBMS_5_Surname.sql

3 Database Description



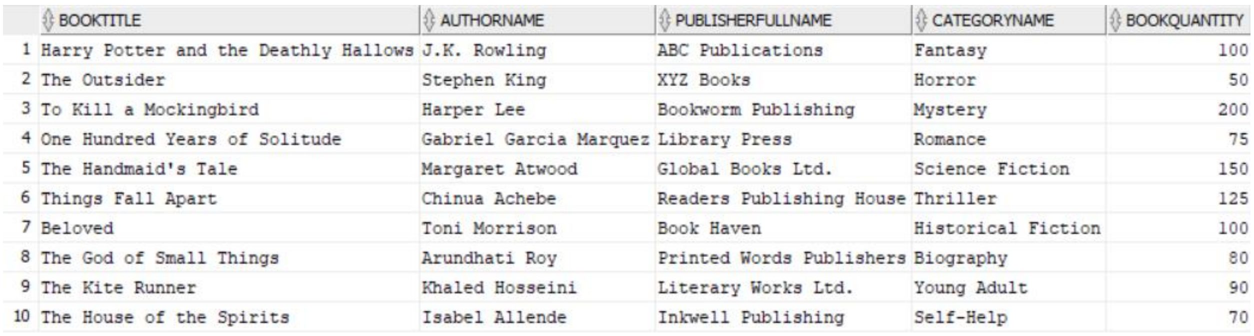
Students
StdID VARCHAR2(9) <pk>
CardNo VARCHAR2(10) <fk>
StdFirsName VARCHAR2(50)
StdLastName VARCHAR2(50)
StdPhoneNo VARCHAR2(12)
StdMail VARCHAR2(50)
StdAddress VARCHAR2(100)
StdPoint NUMBER(3,0)
FK_STUDENTS_STUDENTS__LIBRARYC
FK_LIBRARYC_STUDENTS__STUDENTS
LibraryCards
CardNo VARCHAR2(10) <pk>
CardExpDate DATE
FK_BOOKS_ST_BOOKS_STU_STUDENTS
Books_Students
ISBN VARCHAR2(13) <pk,fk1>
StdID VARCHAR2(9) <pk,fk2>
BookBorrowingDate DATE
BookDeliveryDate DATE
FK_BOOKS_ST_BOOKS_STU_BOOKS
Publishers
PublisherID VARCHAR2(10) <pk>
PublisherFullName VARCHAR2(50)
PublisherAddress VARCHAR2(100)
FK_BOOKS_BOOKS_PUB_PUBLISHE
Books
ISBN VARCHAR2(13) <pk>
AuthorID VARCHAR2(10) <fk2>
PublisherID VARCHAR2(10) <fk1>
BookTitle VARCHAR2(50)
BookPublishDate DATE
BookQuantity SMALLINT
BookPrice REAL
FK_BOOKS_BOOKS_AUT_AUTHORS
Authors
AuthorID VARCHAR2(10) <pk>
AuthorFirstName VARCHAR2(50)
AuthorLastName VARCHAR2(50)
AuthorCountry VARCHAR2(30)
FK_BOOKS_CA_BOOKS_CAT_BOOKS
Books_Categories
ISBN VARCHAR2(13) <pk,fk1>
CategoryID VARCHAR2(5) <pk,fk2>
FK_BOOKS_CA_BOOKS_CAT_CATEGORI
Categories
CategoryID VARCHAR2(5) <pk>
CategoryName VARCHAR2(40)

4 SQL Queries

4.1 *Query #1 BOOK_DETAILS VIEW*

- This view provides detailed information about books, including the book title, author name, publisher name, category names, and the number of available copies.

```
CREATE VIEW BOOK_DETAILS AS
SELECT B.BOOKTITLE, A.AUTHORFIRSTNAME || ' ' || A.AUTHORLASTNAME AS
AUTHORNAME,
      P.PUBLISHERFULLNAME, C.CATEGORYNAME, B.BOOKQUANTITY
FROM BOOKS B
JOIN AUTHORS A ON B.AUTHORID = A.AUTHORID
JOIN PUBLISHERS P ON B.PUBLISHERID = P.PUBLISHERID
JOIN BOOKS_CATEGORIES BC ON B.ISBN = BC.ISBN
JOIN CATEGORIES C ON BC.CATEGORYID = C.CATEGORYID;
```

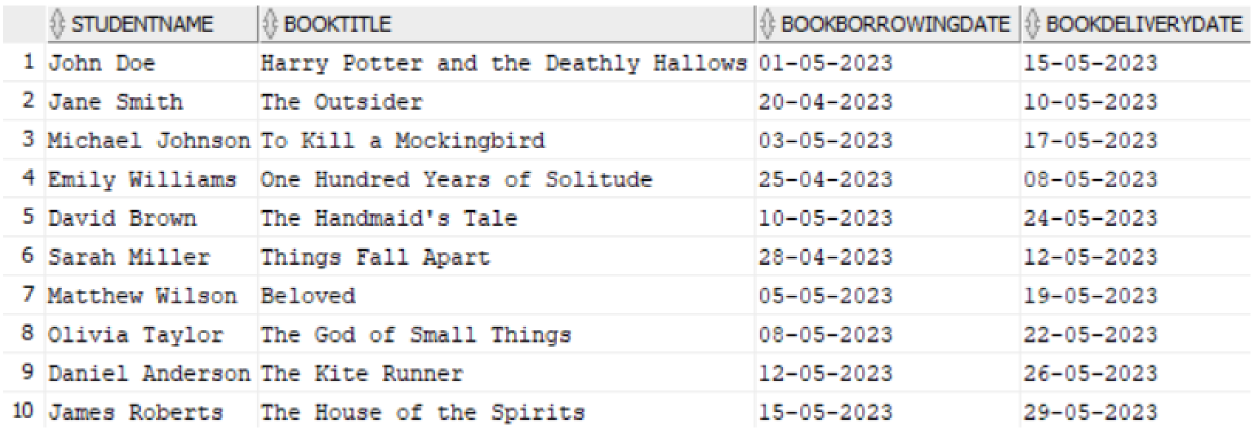


	BOOKTITLE	AUTHORNAME	PUBLISHERFULLNAME	CATEGORYNAME	BOOKQUANTITY
1	Harry Potter and the Deathly Hallows	J.K. Rowling	ABC Publications	Fantasy	100
2	The Outsider	Stephen King	XYZ Books	Horror	50
3	To Kill a Mockingbird	Harper Lee	Bookworm Publishing	Mystery	200
4	One Hundred Years of Solitude	Gabriel Garcia Marquez	Library Press	Romance	75
5	The Handmaid's Tale	Margaret Atwood	Global Books Ltd.	Science Fiction	150
6	Things Fall Apart	Chinua Achebe	Readers Publishing House	Thriller	125
7	Beloved	Toni Morrison	Book Haven	Historical Fiction	100
8	The God of Small Things	Arundhati Roy	Printed Words Publishers	Biography	80
9	The Kite Runner	Khaled Hosseini	Literary Works Ltd.	Young Adult	90
10	The House of the Spirits	Isabel Allende	Inkwell Publishing	Self-Help	70

4.2 *Query #2 STUDENT_BOOKS VIEW*

- This view displays the books borrowed by students along with their names and borrowing details.

```
CREATE VIEW STUDENT_BOOKS AS
SELECT S.STDFIRSNAME || ' ' || S.STDLASTNAME AS STUDENTNAME,
      B.BOOKTITLE, BS.BOOKBORROWINGDATE, BS.BOOKDELIVERYDATE
FROM STUDENTS S
JOIN BOOKS_STUDENTS BS ON S.STDID = BS.STDID
JOIN BOOKS B ON BS.ISBN = B.ISBN;
```

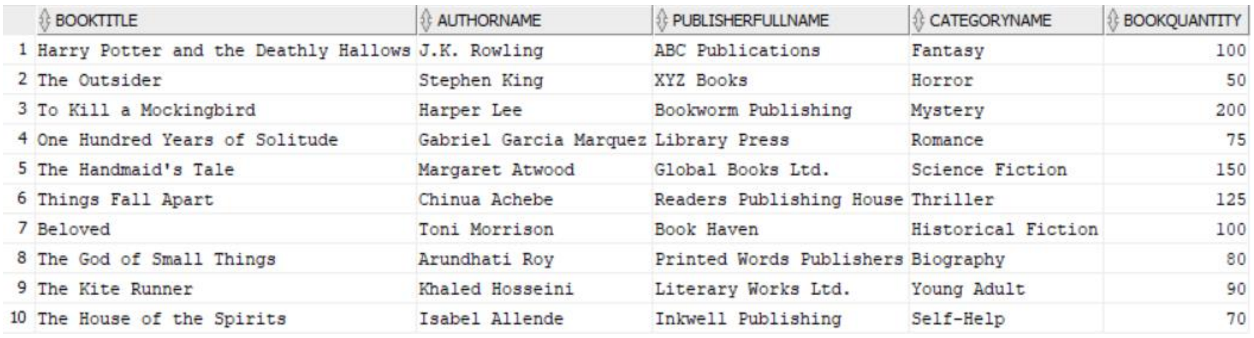


	STUDENTNAME	BOOKTITLE	BOOKBORROWINGDATE	BOOKDELIVERYDATE
1	John Doe	Harry Potter and the Deathly Hallows	01-05-2023	15-05-2023
2	Jane Smith	The Outsider	20-04-2023	10-05-2023
3	Michael Johnson	To Kill a Mockingbird	03-05-2023	17-05-2023
4	Emily Williams	One Hundred Years of Solitude	25-04-2023	08-05-2023
5	David Brown	The Handmaid's Tale	10-05-2023	24-05-2023
6	Sarah Miller	Things Fall Apart	28-04-2023	12-05-2023
7	Matthew Wilson	Beloved	05-05-2023	19-05-2023
8	Olivia Taylor	The God of Small Things	08-05-2023	22-05-2023
9	Daniel Anderson	The Kite Runner	12-05-2023	26-05-2023
10	James Roberts	The House of the Spirits	15-05-2023	29-05-2023

4.3 Query #3 BOOK_DETAILS MATERIALIZED VIEW

- This materialized view provides detailed information about books, including the book title, author name, publisher name, category names, and the number of available copies.

```
CREATE MATERIALIZED VIEW BOOK_DETAILS_MV
REFRESH START WITH SYSDATE NEXT TRUNC(SYSDATE) + 1
AS
SELECT B.BOOKTITLE, A.AUTHORFIRSTNAME || ' ' || A.AUTHORLASTNAME AS
AUTHORNAME,
     P.PUBLISHERFULLNAME, C.CATEGORYNAME, B.BOOKQUANTITY
FROM BOOKS B
JOIN AUTHORS A ON B.AUTHORID = A.AUTHORID
JOIN PUBLISHERS P ON B.PUBLISHERID = P.PUBLISHERID
JOIN BOOKS_CATEGORIES BC ON B.ISBN = BC.ISBN
JOIN CATEGORIES C ON BC.CATEGORYID = C.CATEGORYID;
```

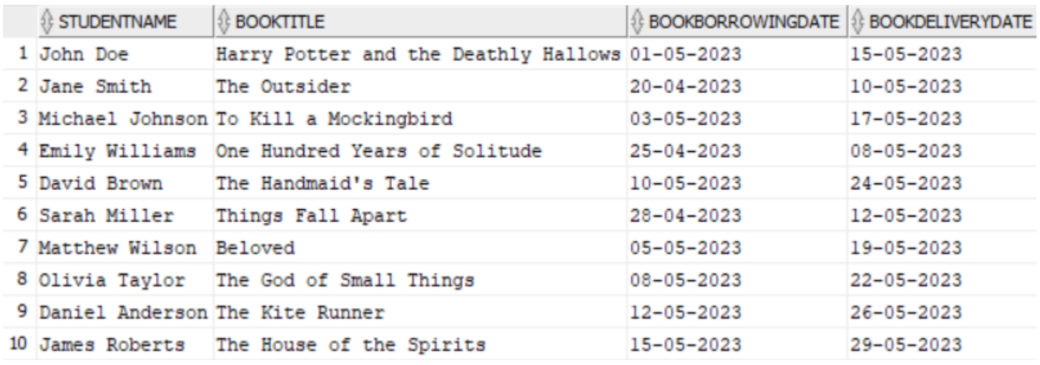


	BOOKTITLE	AUTHORNAME	PUBLISHERFULLNAME	CATEGORYNAME	BOOKQUANTITY
1	Harry Potter and the Deathly Hallows	J.K. Rowling	ABC Publications	Fantasy	100
2	The Outsider	Stephen King	XYZ Books	Horror	50
3	To Kill a Mockingbird	Harper Lee	Bookworm Publishing	Mystery	200
4	One Hundred Years of Solitude	Gabriel Garcia Marquez	Library Press	Romance	75
5	The Handmaid's Tale	Margaret Atwood	Global Books Ltd.	Science Fiction	150
6	Things Fall Apart	Chinua Achebe	Readers Publishing House	Thriller	125
7	Beloved	Toni Morrison	Book Haven	Historical Fiction	100
8	The God of Small Things	Arundhati Roy	Printed Words Publishers	Biography	80
9	The Kite Runner	Khaled Hosseini	Literary Works Ltd.	Young Adult	90
10	The House of the Spirits	Isabel Allende	Inkwell Publishing	Self-Help	70

4.4 Query #4 STUDENT_BOOKS MATERIALIZED VIEW

- This materialized view displays the books borrowed by students along with their names and borrowing details.

```
CREATE MATERIALIZED VIEW STUDENT_BOOKS_MV
REFRESH START WITH SYSDATE NEXT TRUNC(SYSDATE) + 1
AS
SELECT S.STDFIRSNAME || ' ' || S.STDLASTNAME AS STUDENTNAME,
     B.BOOKTITLE, BS.BOOKBORROWINGDATE, BS.BOOKDELIVERYDATE
FROM STUDENTS S
JOIN BOOKS_STUDENTS BS ON S.STDID = BS.STDID
JOIN BOOKS B ON BS.ISBN = B.ISBN;
```

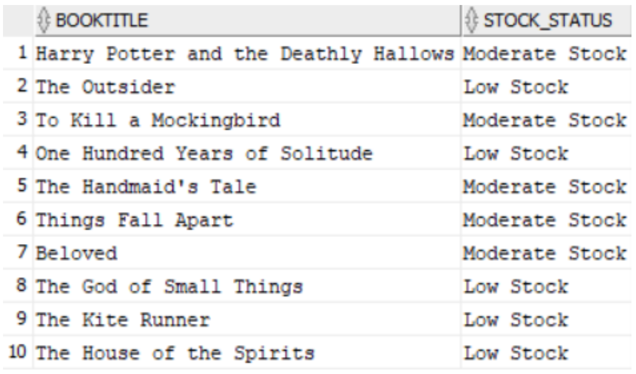


	STUDENTNAME	BOOKTITLE	BOOKBORROWINGDATE	BOOKDELIVERYDATE
1	John Doe	Harry Potter and the Deathly Hallows	01-05-2023	15-05-2023
2	Jane Smith	The Outsider	20-04-2023	10-05-2023
3	Michael Johnson	To Kill a Mockingbird	03-05-2023	17-05-2023
4	Emily Williams	One Hundred Years of Solitude	25-04-2023	08-05-2023
5	David Brown	The Handmaid's Tale	10-05-2023	24-05-2023
6	Sarah Miller	Things Fall Apart	28-04-2023	12-05-2023
7	Matthew Wilson	Beloved	05-05-2023	19-05-2023
8	Olivia Taylor	The God of Small Things	08-05-2023	22-05-2023
9	Daniel Anderson	The Kite Runner	12-05-2023	26-05-2023
10	James Roberts	The House of the Spirits	15-05-2023	29-05-2023

4.5 Query #5 CASE EXPRESSION 1

- In this query, the CASE expression is used to determine the stock status of each book based on its quantity (BOOKQUANTITY column) in the BOOKS table. The CASE expression evaluates the value of BOOKQUANTITY and assigns a corresponding label to the STOCK_STATUS column.

```
SELECT B.BOOKTITLE,
     CASE
        WHEN B.BOOKQUANTITY = 0 THEN 'Out of Stock'
        WHEN B.BOOKQUANTITY > 0 AND B.BOOKQUANTITY <= 90 THEN 'Low Stock'
        WHEN B.BOOKQUANTITY > 90 AND B.BOOKQUANTITY <= 200 THEN 'Moderate
Stock'
        ELSE 'High Stock'
     END AS STOCK_STATUS
FROM BOOKS B;
```

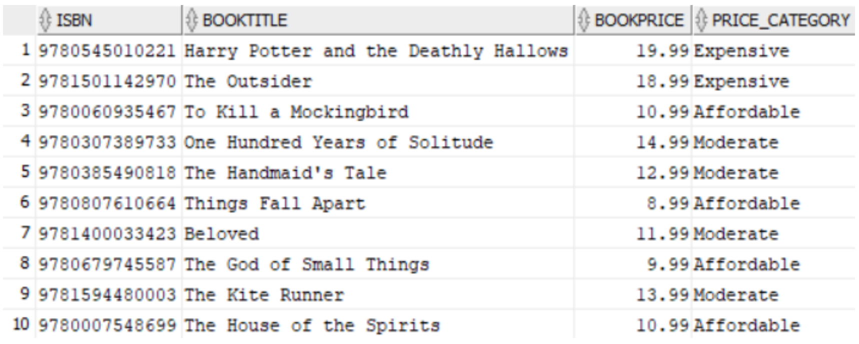


	BOOKTITLE	STOCK_STATUS
1	Harry Potter and the Deathly Hallows	Moderate Stock
2	The Outsider	Low Stock
3	To Kill a Mockingbird	Moderate Stock
4	One Hundred Years of Solitude	Low Stock
5	The Handmaid's Tale	Moderate Stock
6	Things Fall Apart	Moderate Stock
7	Beloved	Moderate Stock
8	The God of Small Things	Low Stock
9	The Kite Runner	Low Stock
10	The House of the Spirits	Low Stock

4.6 Query #6 CASE EXPRESSION 2

- In this query, the CASE expression is used to categorize the books based on their prices (BOOKPRICE column) in the BOOKS table. The CASE expression evaluates the value of BOOKPRICE and assigns a corresponding label to the PRICE_CATEGORY column.

```
SELECT B.ISBN,
     B.BOOKTITLE,
     B.BOOKPRICE,
     CASE
        WHEN B.BOOKPRICE > 16 THEN 'Expensive'
        WHEN B.BOOKPRICE > 11 THEN 'Moderate'
        ELSE 'Affordable'
     END AS PRICE_CATEGORY
FROM BOOKS B;
```

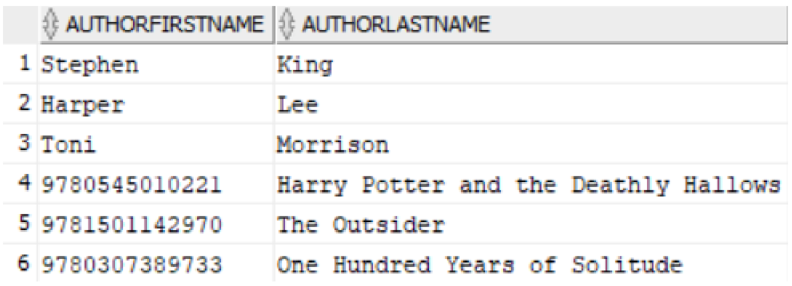


	ISBN	BOOKTITLE	BOOKPRICE	PRICE_CATEGORY
1	9780545010221	Harry Potter and the Deathly Hallows	19.99	Expensive
2	9781501142970	The Outsider	18.99	Expensive
3	9780060935467	To Kill a Mockingbird	10.99	Affordable
4	9780307389733	One Hundred Years of Solitude	14.99	Moderate
5	9780385490818	The Handmaid's Tale	12.99	Moderate
6	9780807610664	Things Fall Apart	8.99	Affordable
7	9781400033423	Beloved	11.99	Moderate
8	9780679745587	The God of Small Things	9.99	Affordable
9	9781594480003	The Kite Runner	13.99	Moderate
10	9780007548699	The House of the Spirits	10.99	Affordable

4.7 Query #7 UNION ALL

- This is a query that combines the results of two separate queries using the UNION ALL operator.

```
SELECT A.AUTHORFIRSTNAME, A.AUTHORLASTNAME
FROM AUTHORS A
WHERE A.AUTHORCOUNTRY = 'United States'
UNION ALL
SELECT B.ISBN, B.BOOKTITLE
FROM BOOKS B
WHERE B.BOOKPRICE > 14;
```

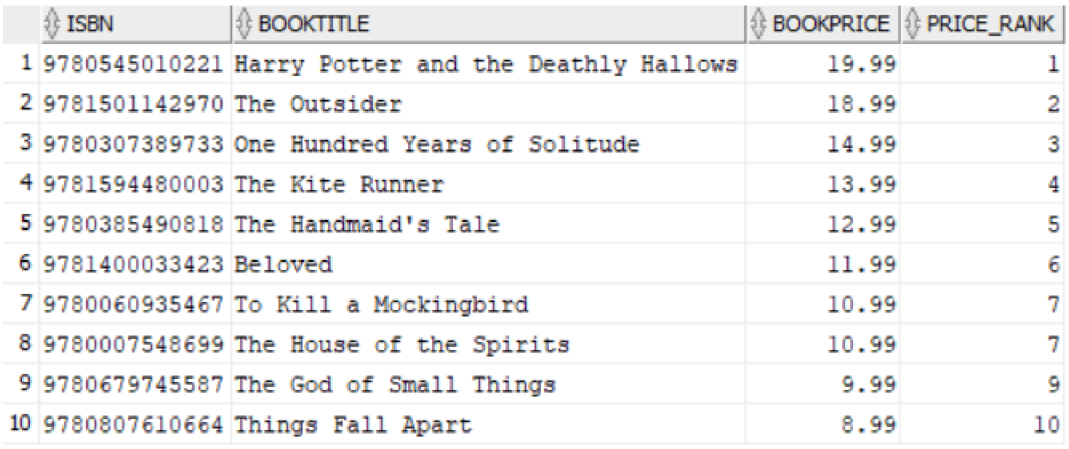


	AUTHORFIRSTNAME	AUTHORLASTNAME
1	Stephen	King
2	Harper	Lee
3	Toni	Morrison
4	9780545010221	Harry Potter and the Deathly Hallows
5	9781501142970	The Outsider
6	9780307389733	One Hundred Years of Solitude

4.8 Query #8 ANALYTICAL FUNCTION 1

- In this query, the RANK function is used to assign a rank to each book based on its price (BOOKPRICE column) in descending order. The ORDER BY B.BOOKPRICE DESC clause specifies the ordering based on the book price. The PRICE_RANK column will contain the rank assigned to each book.

```
SELECT B.ISBN,
       B.BOOKTITLE,
       B.BOOKPRICE,
       RANK() OVER (ORDER BY B.BOOKPRICE DESC) AS PRICE_RANK
FROM BOOKS B;
```

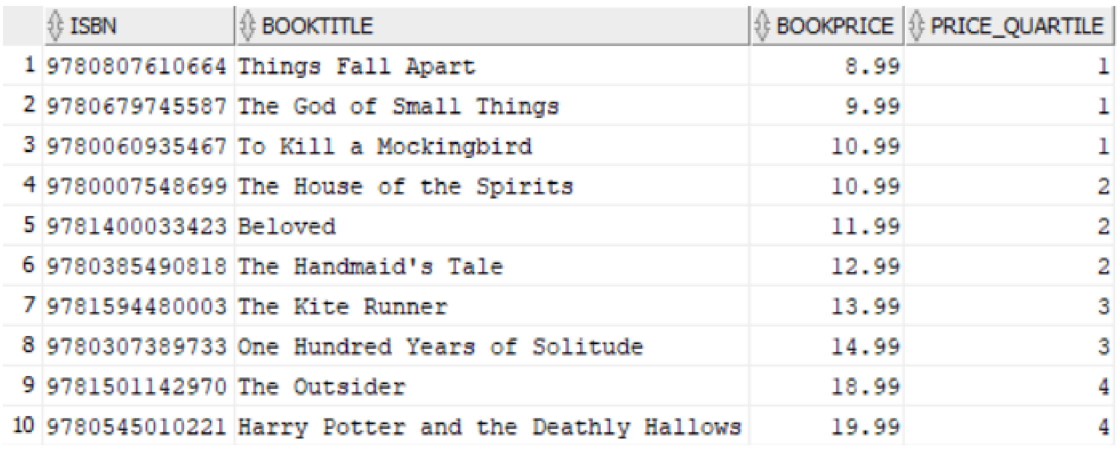


	ISBN	BOOKTITLE	BOOKPRICE	PRICE_RANK
1	9780545010221	Harry Potter and the Deathly Hallows	19.99	1
2	9781501142970	The Outsider	18.99	2
3	9780307389733	One Hundred Years of Solitude	14.99	3
4	9781594480003	The Kite Runner	13.99	4
5	9780385490818	The Handmaid's Tale	12.99	5
6	9781400033423	Beloved	11.99	6
7	9780060935467	To Kill a Mockingbird	10.99	7
8	9780007548699	The House of the Spirits	10.99	7
9	9780679745587	The God of Small Things	9.99	9
10	9780807610664	Things Fall Apart	8.99	10

4.9 Query #9 ANALYTICAL FUNCTION 2

- In this query, the NTILE function is used to divide the books into quartiles based on their price (BOOKPRICE column) in ascending order. The ORDER BY B.BOOKPRICE clause specifies the ordering based on the book price. The NTILE(4) function divides the data into four equal groups. The PRICE_QUARTILE column will contain the quartile number assigned to each book.

```
SELECT B.ISBN,
       B.BOOKTITLE,
       B.BOOKPRICE,
       NTILE(4) OVER (ORDER BY B.BOOKPRICE) AS PRICE_QUARTILE
FROM BOOKS B;
```

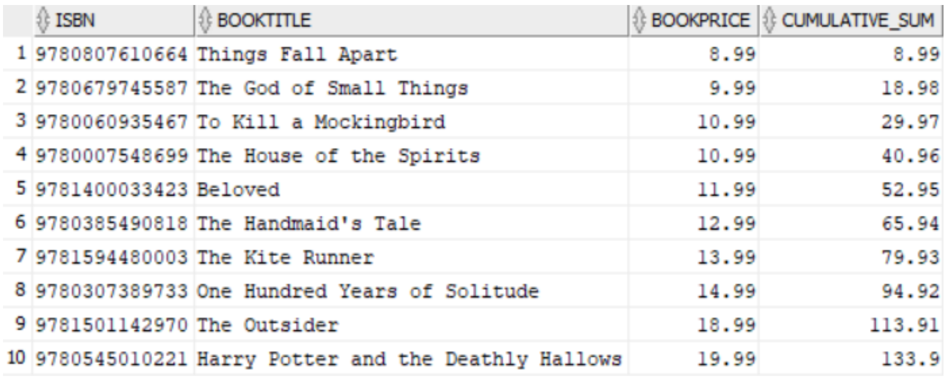


	ISBN	BOOKTITLE	BOOKPRICE	PRICE_QUARTILE
1	9780807610664	Things Fall Apart	8.99	1
2	9780679745587	The God of Small Things	9.99	1
3	9780060935467	To Kill a Mockingbird	10.99	1
4	9780007548699	The House of the Spirits	10.99	2
5	9781400033423	Beloved	11.99	2
6	9780385490818	The Handmaid's Tale	12.99	2
7	9781594480003	The Kite Runner	13.99	3
8	9780307389733	One Hundred Years of Solitude	14.99	3
9	9781501142970	The Outsider	18.99	4
10	9780545010221	Harry Potter and the Deathly Hallows	19.99	4

4.10 Query #10 WINDOWING FUNCTION 1

- In this query, the SUM function is used as a window function to calculate the cumulative sum of book prices (BOOKPRICE column) based on the book's ID (ISBN column). The ORDER BY B.BOOKPRICE clause ensures that the rows are ordered by book ID. The ROWS BETWEEN UNBOUNDED PRECEDING AND CURRENT ROW clause defines the window frame, indicating that the sum should be calculated from the start of the partition (unbounded preceding) up to the current row.

```
SELECT B.ISBN,
       B.BOOKTITLE,
       B.BOOKPRICE,
       SUM(B.BOOKPRICE) OVER (ORDER BY B.BOOKPRICE ROWS BETWEEN
UNBOUNDED PRECEDING AND CURRENT ROW) AS CUMULATIVE_SUM
FROM BOOKS B;
```

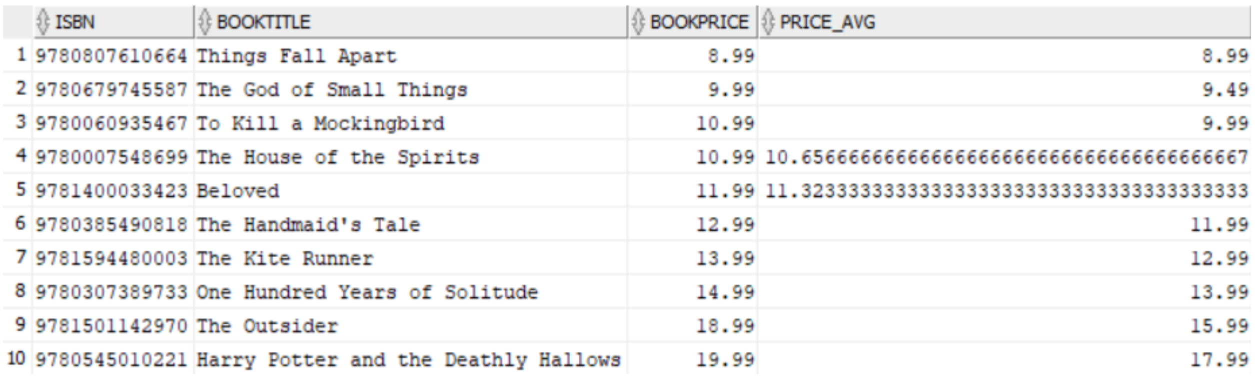


	ISBN	BOOKTITLE	BOOKPRICE	CUMULATIVE_SUM
1	9780807610664	Things Fall Apart	8.99	8.99
2	9780679745587	The God of Small Things	9.99	18.98
3	9780060935467	To Kill a Mockingbird	10.99	29.97
4	9780007548699	The House of the Spirits	10.99	40.96
5	9781400033423	Beloved	11.99	52.95
6	9780385490818	The Handmaid's Tale	12.99	65.94
7	9781594480003	The Kite Runner	13.99	79.93
8	9780307389733	One Hundred Years of Solitude	14.99	94.92
9	9781501142970	The Outsider	18.99	113.91
10	9780545010221	Harry Potter and the Deathly Hallows	19.99	133.9

4.11 Query #11 WINDOWING FUNCTION 2

- In this query, the AVG function is used as a windowing function to calculate the average book price (BOOKPRICE column) within a fixed window size. The ORDER BY B.BOOKPRICE clause specifies the ordering based on the book price. The ROWS BETWEEN 2 PRECEDING AND CURRENT ROW clause defines the window as including the current row and the two preceding rows. The PRICE_AVG column will contain the average book price within the window for each row.

```
SELECT B.ISBN,
       B.BOOKTITLE,
       B.BOOKPRICE,
       AVG(B.BOOKPRICE) OVER (ORDER BY B.BOOKPRICE ROWS BETWEEN 2
PRECEDING AND CURRENT ROW) AS PRICE_AVG
FROM BOOKS B;
```

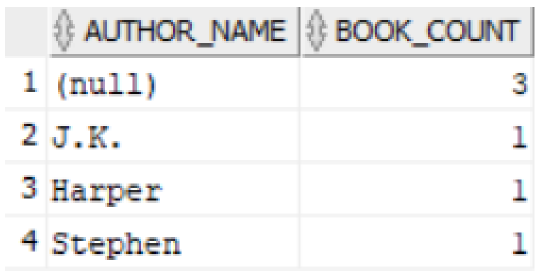


	ISBN	BOOKTITLE	BOOKPRICE	PRICE_AVG
1	9780807610664	Things Fall Apart	8.99	8.99
2	9780679745587	The God of Small Things	9.99	9.49
3	9780060935467	To Kill a Mockingbird	10.99	9.99
4	9780007548699	The House of the Spirits	10.99	10.656666666666666666666666666666666667
5	9781400033423	Beloved	11.99	11.323333333333333333333333333333333333
6	9780385490818	The Handmaid's Tale	12.99	11.99
7	9781594480003	The Kite Runner	13.99	12.99
8	9780307389733	One Hundred Years of Solitude	14.99	13.99
9	9781501142970	The Outsider	18.99	15.99
10	9780545010221	Harry Potter and the Deathly Hallows	19.99	17.99

4.12 Query #12 DIMENSION TABLES AND FACT TABLE

- In this query, we use the GROUP BY CUBE clause to generate a result set that includes subtotals and grand totals for all combinations of author and publisher. The COUNT(*) function is used to calculate the number of books (BOOK_COUNT) in each combination.

```
CREATE TABLE DIM_AUTHOR (
  AUTHOR_ID        VARCHAR2(10) PRIMARY KEY,
  AUTHOR_NAME      VARCHAR2(30)
);

CREATE TABLE DIM_PUBLISHER (
  PUBLISHER_ID     VARCHAR2(10) PRIMARY KEY,
  PUBLISHER_NAME   VARCHAR2(30)
);

CREATE TABLE DIM_GENRE (
  GENRE_ID         VARCHAR2(10) PRIMARY KEY,
  GENRE_NAME       VARCHAR2(30)
);

CREATE TABLE FACT_BOOK (
  BOOK_ID          VARCHAR2(10) PRIMARY KEY,
  AUTHOR_ID        VARCHAR2(10),
  PUBLISHER_ID     VARCHAR2(10),
  GENRE_ID        VARCHAR2(10),
  BOOK_TITLE       VARCHAR2(50),
  BOOK_PRICE       DECIMAL(10, 2),
  FOREIGN KEY (AUTHOR_ID) REFERENCES DIM_AUTHOR(AUTHOR_ID),
  FOREIGN KEY (PUBLISHER_ID) REFERENCES DIM_PUBLISHER(PUBLISHER_ID),
  FOREIGN KEY (GENRE_ID) REFERENCES DIM_GENRE(GENRE_ID)
);

INSERT INTO DIM_AUTHOR (AUTHOR_ID, AUTHOR_NAME) VALUES ('A01', 'J.K.');
INSERT INTO DIM_AUTHOR (AUTHOR_ID, AUTHOR_NAME) VALUES ('A02', 'Stephen');
INSERT INTO DIM_AUTHOR (AUTHOR_ID, AUTHOR_NAME) VALUES ('A03', 'Harper');

INSERT INTO DIM_PUBLISHER (PUBLISHER_ID, PUBLISHER_NAME) VALUES ('PUB01', 'ABC
Publications');
INSERT INTO DIM_PUBLISHER (PUBLISHER_ID, PUBLISHER_NAME) VALUES ('PUB02', 'XYZ
Books');
INSERT INTO DIM_PUBLISHER (PUBLISHER_ID, PUBLISHER_NAME) VALUES ('PUB03',
'Bookworm Publishing');

INSERT INTO DIM_GENRE (GENRE_ID, GENRE_NAME) VALUES ('C01', 'Fantasy');
INSERT INTO DIM_GENRE (GENRE_ID, GENRE_NAME) VALUES ('C02', 'Horror');
INSERT INTO DIM_GENRE (GENRE_ID, GENRE_NAME) VALUES ('C03', 'Mystery');

INSERT INTO FACT_BOOK (BOOK_ID, AUTHOR_ID, PUBLISHER_ID, GENRE_ID,
BOOK_TITLE, BOOK_PRICE) VALUES ('C01', 'A01', 'PUB01', 'C01', 'Harry Potter and the Deathly
Hallows', 19.99);
INSERT INTO FACT_BOOK (BOOK_ID, AUTHOR_ID, PUBLISHER_ID, GENRE_ID,
BOOK_TITLE, BOOK_PRICE) VALUES ('C02', 'A02', 'PUB02', 'C02', 'The Outsider', 18.99);
INSERT INTO FACT_BOOK (BOOK_ID, AUTHOR_ID, PUBLISHER_ID, GENRE_ID,
BOOK_TITLE, BOOK_PRICE) VALUES ('C03', 'A03', 'PUB03', 'C03', 'To Kill a Mockingbird', 10.99);

SELECT DA.AUTHOR_NAME, COUNT(*) AS BOOK_COUNT
FROM FACT_BOOK FB
JOIN DIM_AUTHOR DA ON FB.AUTHOR_ID = DA.AUTHOR_ID
GROUP BY CUBE (DA.AUTHOR_NAME);
```

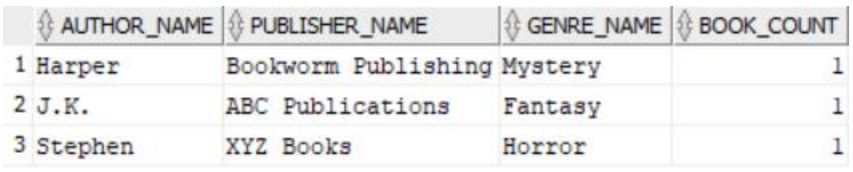


	AUTHOR_NAME	BOOK_COUNT
1	(null)	3
2	J.K.	1
3	Harper	1
4	Stephen	1

4.13 Query #13 MODEL, PARTITION BY, DIMENSION BY

- In this query, we first perform a regular grouping operation to calculate the book count for each combination of author, publisher, and genre. Then, we use the MODEL clause to apply additional calculations and transformations.

```
SELECT AUTHOR_NAME, PUBLISHER_NAME, GENRE_NAME, BOOK_COUNT
FROM (
  SELECT DA.AUTHOR_NAME, DP.PUBLISHER_NAME, DG.GENRE_NAME, COUNT(*)
AS BOOK_COUNT
  FROM FACT_BOOK FB
  JOIN DIM_AUTHOR DA ON FB.AUTHOR_ID = DA.AUTHOR_ID
  JOIN DIM_PUBLISHER DP ON FB.PUBLISHER_ID = DP.PUBLISHER_ID
  JOIN DIM_GENRE DG ON FB.GENRE_ID = DG.GENRE_ID
  GROUP BY DA.AUTHOR_NAME, DP.PUBLISHER_NAME, DG.GENRE_NAME
)
MODEL
  PARTITION BY (AUTHOR_NAME)
  DIMENSION BY (PUBLISHER_NAME, GENRE_NAME)
  MEASURES (BOOK_COUNT)
  RULES (
    BOOK_COUNT[ANY, ANY] = SUM(BOOK_COUNT)[PUBLISHER_NAME,
GENRE_NAME]
);
```



	AUTHOR_NAME	PUBLISHER_NAME	GENRE_NAME	BOOK_COUNT
1	Harper	Bookworm Publishing	Mystery	1
2	J.K.	ABC Publications	Fantasy	1
3	Stephen	XYZ Books	Horror	1

5 Conclusions

Criteria	Minimum	Count of Queries	Number of Queries
Creating views	2	2	4.1, 4.2
Creating materialized views	2	2	4.3, 4.4
Queries with CASE expression	2	2	4.5, 4.6
Hierarchical query using START WITH, CONNECT BY and LEVEL	1	2	4.3, 4.4
Hierarchical query using WITH, UNION ALL	1	1	4.7
Analytical functions	2	2	4.8, 4.9
Windowing functions	2	2	4.10, 4.11
GROUP BY CUBE, GROUPING(), GROUPING SETS	2	1	4.12
MODEL, PARTITION BY, DIMENSION BY, MEASURES, RULES	2	1	4.134.13

In this task, I explored various aspects of SQL queries and database operations. I started by creating views and materialized views to provide different perspectives on the data. Then I used the CASE expression to perform conditional operations and analytical functions to derive insights from the dataset. Additionally, I utilized windowing functions to define specific ranges for data analysis. Finally, I designed dimension and fact tables and applied grouping operations using GROUP BY CUBE. These techniques allow for flexible data manipulation and analysis, enabling us to extract valuable information from the database.

6 References

- Riga Technical University, Faculty of Computer Science and Information Technology, Institute of Applied Computer Systems, DSP201 – Database Management Systems, Presentations