REPORT

Introduction:
The purpose of the report is to describe the database system for book store. The database system is intended to be used to manage their storage, sales, and users data. The system include entities such as books,users,authors,publisher,order,reviews,payment.
ER Diagram:

This diagram of the database system follows the normal forms. The first normal form (1NF) requires that each column in a table must contain atomic values. In our design, each table contains atomic values, and no column contains repeating groups of data.The second normal form (2NF) requires that each non-key attribute is fully dependent on the primary key. In our design, each non-key attribute depends on the primary key. Finally, the third normal form (3NF) requires that there is no transitive dependency between non-key attributes. In our design, there are no transitive dependencies between non-key attributes.

Explanation of 'add following' part:

```
1."CREATE OR REPLACE PROCEDURE group_by_infromation
   is
   begin
      for book in (Select genre, AVG(price) as avg_price,count(*) as num_books from books
         group by genre) loop
         dbms_output.put_line('Genre: ' || book.genre || '- Average Price: ' || book.avg_price || '-
Number of books: ' || book.num_books);
      end loop;
   end;
"
```

this procedure is group all books by genre and output average price of all books on this genre and number of books in database with this genre

2.

```
CREATE OR REPLACE function numOfrec(table_name in varchar)
   return integer is num integer;
   begin
      execute immediate 'SELECT count(*) from ' || table_name into num;
      return num;
   end;
```

This function is using for find the number of records of certain table in parameter by executing SQL query the result will be integer .
3.

```
CREATE OR REPLACE PROCEDURE update_proccess is
   BEGIN
      update payment
```

```
    set status='ready for confirmation'
    where order_id in
      (SELECT order_id from orders
      where sysdate-dateoforder>3);
    update payment
    set status='rejected'
    where order_id in
      (select order_id from orders
      where sysdate-dateoforder>7);

    dbms_output.put_line(sql%rowcount);
  end;
```

This procedure is using to update status of payment table. After 3 days all data processed user should confirm , if after 7 days the data won't confirm it automatically should reject the order.in the end, procedure output rows effected.

```
4.CREATE OR REPLACE PROCEDURE add_book(p_name varchar, p_genre varchar, p_price
varchar, p_publisher_id number, p_author_id number) AS
    len integer;
    notokpub exception;
    notokauth exception;
    notoklen exception;
    auth_id number;
    pub_id number;
  BEGIN
    len := LENGTH(p_name);
    IF len < 5 THEN
      raise notoklen;
    END IF;
    select author_id into auth_id from authors where author_id = p_author_id;
    select publisher_id into pub_id from publishers where publisher_id = p_publisher_id;
    if auth_id is null then
      raise notokauth;
    end if;
    if pub_id is null then
      raise notokpub;
    end if;
    INSERT INTO books(name, genre, price, publisher_id, author_id)
    VALUES (p_name, p_genre, p_price, p_publisher_id, p_author_id);

    DBMS_OUTPUT.PUT_LINE('Book added successfully!');
    commit;
  EXCEPTION
    when notoklen then
      rollback;
```

```
            dbms_output.put_line('Book should have at least 5 character');
        when notokauth then
            rollback;
            dbms_output.put_line('There is no author with that id');
        when notokpub then
            rollback;
            dbms_output.put_line('There is no publisher with that id');
    END;
```

this procedure is using to add book to the database. There is 3 user-defined exceptions. Each exception controlling this statements :book should have at least 5 character , and due to author_id and publisher_id are referencing from another table. So firstly it checks for the correction of the parameters , if there is no mistake then . it will inform about it, or if there will be mistake it will inform where this mistake was made.

```
create trigger trigger_review
    before insert on reviews
    referencing old as o new as n
    for each row
    declare
        numOfrows integer;
    begin
        UPDATE Books
        SET rating = (SELECT avg(rating) FROM Reviews
                    WHERE book_id =: n.book_id)
        WHERE book_id =: n.book_id;
        select count(*) into numOfrows from Reviews;
        dbms_output.put_line(numOfrows);
    end;
```

If Any user want to review some book , before it system will show update rating of the book and show number of reviews was before this review, to show some feedback of other users to this book;