Учреждение образования «БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ  
ТЕХНОЛОГИЧЕСКИЙ УНИВЕРСИТЕТ»

Факультет информационных технологий

[Кафедра информационных](https://www.belstu.by/fakultety/fit/vm) систем и технологий

Специальность 1-40 01 01 Программное обеспечение информационных технологий

**Отчёт по лабораторной работе №2**

по дисциплине Проектирование и разработка баз данных интернет-приложений

Тема: Реализация базы данных

Исполнитель:

Студент 3 курса группы 6

Русак Наталья Александровна

Руководитель:

Ассистент Нистюк О. А.

Минск, 2024

1. **Список таблиц в СУБД MS SQL Server**

**Таблица AUTHOR**

|  |  |  |  |
| --- | --- | --- | --- |
| Название атрибута | Тип данных | Ограничения | Назначение |
| ID | INT | Primary key, identity | Уникальный идентификатор автора |
| FIRST\_NAME | NVARCHAR(50) | Not null | Имя автора |
| LAST\_NAME | NVARCHAR(50) | Not null | Фамилия автора |
| PATRONYMIC | NVARCHAR(50) | Not null | Отчество автора |
| COUNTRY | NVARCHAR(50) | Not null | Страна происхождения автора |

**Таблица GENRE**

|  |  |  |  |
| --- | --- | --- | --- |
| Название атрибута | Тип данных | Ограничения | Назначение |
| ID | INT | Primary key, identity | Уникальный идентификатор жанра |
| NAME | NVARCHAR(50) | Not null | Название жанра |
| AGE\_RATING | NVARCHAR(10) | Not null | Возрастной рейтинг жанра |

**Таблица BOOK**

|  |  |  |  |
| --- | --- | --- | --- |
| Название атрибута | Тип данных | Ограничения | Назначение |
| ID | INT | Primary key, identity | Уникальный идентификатор книги |
| AUTHOR\_ID | INT | Not null, foreign key | Внешний ключ на таблицу AUTHOR |
| GENRE\_ID | INT | Not null, foreign key | Внешний ключ на таблицу GENRE |
| NAME | NVARCHAR(100) | Not null | Название книги |
| YEAR\_OF\_PUBLISHING | INT |  | Год публикации книги |

**Таблица CUSTOMER**

|  |  |  |  |
| --- | --- | --- | --- |
| Название атрибута | Тип данных | Ограничения | Назначение |
| ID | INT | Primary key, identity | Уникальный идентификатор клиента |
| FIRST\_NAME | NVARCHAR(70) | Not null | Имя заказчика |
| LAST\_NAME | NVARCHAR(70) | Not null | Фамилия заказчика |
| PATRONYMIC | NVARCHAR(70) | Not null | Отчество заказчика |
| PHONE\_NUMBER | VARCHAR(20) | Not null | Номер телефона заказчика |
| EMAIL | VARCHAR(70) | Not null | Электронная почта заказчика |
| TYPE | NVARCHAR(20) | CHECK (TYPE IN (N'Юридическое лицо', N'Физическое лицо')) | Тип заказчика (юридическое или физическое лицо) |

**Таблица BOOK\_CHARACTERISTICS**

|  |  |  |  |
| --- | --- | --- | --- |
| Название атрибута | Тип данных | Ограничения | Назначение |
| ID | INT | Primary key, identity | Уникальный идентификатор характеристики книги |
| COVER\_TYPE | NVARCHAR(50) | Not null | Тип обложки книги |
| PAPER\_TYPE | NVARCHAR(50) | Not null | Тип бумаги |
| BINDING\_TYPE | NVARCHAR(50) | Not null | Тип переплета |
| NUMBER\_OF\_PAGES | INT | Not null | Количество страниц |
| PAPER\_SIZE | NVARCHAR(50) | Not null | Размер бумаги |

**Таблица BOOK\_ORDER**

|  |  |  |  |
| --- | --- | --- | --- |
| Название атрибута | Тип данных | Ограничения | Назначение |
| ID | INT | Primary key, identity | Уникальный идентификатор заказа |
| BOOK\_ID | INT | Not null, foreign key | Внешний ключ на таблицу BOOK |
| CUSTOMER\_ID | INT | Not null, foreign key | Внешний ключ на таблицу CUSTOMER |
| BOOK\_CHARACTERISTICS\_ID | INT | Not null, foreign key | Внешний ключ на таблицу BOOK\_CHARACTERISTICS |
| DATE\_OF\_ORDER | DATE | Not null | Дата заказа книги |
| EDITION | INT | Not null | Тираж книги |
| TOTAL\_PRICE | DECIMAL(10, 2) | Not null | Общая цена заказа |
| STATUS | VARCHAR(20) | CHECK (STATUS IN (N'Создан', N'В работе', N'Готов')) | Статус заказа (создан, в работе, готов) |

1. **Прочие объекты в СУБД MS SQL Server**

**BOOK\_DETAILS:**

* Тип: представление.
* Назначение: предоставляет детальную информацию о книгах, включая данные об авторе и жанре.
* Зависимые объекты: зависит от таблиц BOOK, AUTHOR и GENRE.

**CUSTOMER\_ORDER\_SUMMARY:**

* Тип: представление.
* Назначение: предоставляет сводку о заказах клиентов, включая количество заказов и общую сумму заказов.
* Зависимые объекты: зависит от таблиц CUSTOMER и BOOK\_ORDER.

**ORDER\_DETAILS:**

* Тип: представление.
* Назначение: предоставляет детальную информацию о заказах книг, включая данные о книге, авторе, жанре, характеристиках книги и статусе заказа.
* Зависимые объекты: зависит от таблиц BOOK\_ORDER, BOOK, AUTHOR, GENRE и BOOK\_CHARACTERISTICS.

**IDX\_BOOK\_NAME:**

* Тип: индекс.
* Назначение: индексирует столбец NAME в таблице BOOK для повышения производительности при выполнении запросов, которые используют сортировки по этому полю.

**IDX\_AUTHOR\_NAME:**

* Тип: индекс.
* Назначение: индексирует столбцы LAST\_NAME, FIRST\_NAME и PATRONYMIC в таблице AUTHOR для повышения производительности при выполнении запросов, которые используют эти поля.

**IDX\_ORDER\_DATE:**

* Тип: индекс.
* Назначение: индексирует столбец DATE\_OF\_ORDER в таблице BOOK\_ORDER для повышения производительности при выполнении запросов, использующих сортировки по этому полю.

Скрипт для создания объектов приведен в листинге 2.1.

|  |
| --- |
| CREATE VIEW BOOK\_DETAILS AS  SELECT b.ID,  b.NAME AS BOOK\_NAME,  a.FIRST\_NAME,  a.LAST\_NAME,  a.PATRONYMIC,  g.NAME AS GENRE\_NAME,  g.AGE\_RATING  FROM BOOK b  INNER JOIN AUTHOR a ON b.AUTHOR\_ID = a.ID  INNER JOIN GENRE g ON b.GENRE\_ID = g.ID;  CREATE VIEW CUSTOMER\_ORDER\_SUMMARY AS  SELECT c.ID,  c.FIRST\_NAME,  c.LAST\_NAME,  c.PATRONYMIC,  COUNT(o.ID) AS TOTAL\_ORDERS,  SUM(o.TOTAL\_PRICE) AS TOTAL\_PRICE  FROM CUSTOMER c  LEFT JOIN BOOK\_ORDER o ON c.ID = o.CUSTOMER\_ID  GROUP BY c.ID, c.FIRST\_NAME, c.LAST\_NAME, c.PATRONYMIC;  CREATE VIEW ORDER\_DETAILS AS  SELECT o.ID , b.NAME AS BOOK\_NAME,  a.FIRST\_NAME,  a.LAST\_NAME,  a.PATRONYMIC,  g.NAME AS GENRE,  bc.COVER\_TYPE, bc.PAPER\_TYPE,  bc.BINDING\_TYPE, bc.NUMBER\_OF\_PAGES,  bc.PAPER\_SIZE,  o.DATE\_OF\_ORDER, o.EDITION,  o.TOTAL\_PRICE, o.STATUS  FROM BOOK\_ORDER o  INNER JOIN BOOK b ON o.BOOK\_ID = b.ID  INNER JOIN AUTHOR a ON b.AUTHOR\_ID = a.ID  INNER JOIN GENRE g ON b.GENRE\_ID = g.ID  INNER JOIN BOOK\_CHARACTERISTICS bc ON o.BOOK\_CHARACTERISTICS\_ID = bc.ID;  CREATE INDEX IDX\_BOOK\_NAME ON BOOK (NAME);  CREATE INDEX IDX\_AUTHOR\_NAME ON AUTHOR (LAST\_NAME, FIRST\_NAME, PATRONYMIC);  CREATE INDEX IDX\_ORDER\_DATE ON BOOK\_ORDER (DATE\_OF\_ORDER); |

Листинг 2.1 – Скрипт для создания представлений и индексов

1. **Хранимые процедуры в СУБД MS SQL Server**

**Хранимая процедура ADD\_BOOK:**

* Параметры:
* @Name (NVARCHAR(100)): Название книги.
* @AuthorID (INT): Идентификатор автора книги.
* @YearOfPublishing (INT): Год публикации книги.
* @GenreID (INT): Идентификатор жанра книги.
* Назначение: добавляет новую книгу в базу данных.
* Зависимости: использует таблицу BOOK и проверяет наличие автора и жанра в таблицах AUTHOR и GENRE.

**Хранимая процедура DELETE\_BOOK:**

* Параметры:
* @BookID (INT): Идентификатор книги для удаления.
* Назначение: удаляет книгу из базы данных.
* Зависимости: использует таблицу BOOK.

**Хранимая процедура GET\_CUSTOMER\_ORDERS:**

* Параметры:
* @CustomerID (INT): Идентификатор клиента.
* Назначение: получает заказы клиента по его идентификатору.
* Зависимости: использует таблицу CUSTOMER и BOOK\_ORDER.

**Хранимая процедура UPDATE\_ORDER\_STATUS:**

* Параметры:
* @OrderID (INT): Идентификатор заказа.
* @NewStatus (VARCHAR(20)): Новый статус заказа.
* Назначение: обновляет статус заказа.
* Зависимости: использует таблицу BOOK\_ORDER.

Скрипт для создания хранимых процедур приведен в листинге 3.1.

|  |
| --- |
| CREATE PROCEDURE ADD\_BOOK @Name NVARCHAR(100),  @AuthorID INT,  @YearOfPublishing INT,  @GenreID INT  AS  BEGIN  IF NOT EXISTS (SELECT 1 FROM AUTHOR WHERE ID = @AuthorID)  BEGIN  RAISERROR (N'Автор с указанным ID не существует.', 16, 1);  END  INSERT INTO BOOK (NAME, AUTHOR\_ID, YEAR\_OF\_PUBLISHING, GENRE\_ID)  VALUES (@Name, @AuthorID, @YearOfPublishing, @GenreID);  END;  CREATE PROCEDURE DELETE\_BOOK @BookID INT  AS  BEGIN  IF NOT EXISTS (SELECT 1 FROM BOOK WHERE ID = @BookID)  BEGIN  RAISERROR (N'Книга с указанным ID не существует.', 16, 1);  END  DELETE FROM BOOK WHERE ID = @BookID;  END;  CREATE PROCEDURE GET\_CUSTOMER\_ORDERS  @CustomerID INT  AS  BEGIN  IF NOT EXISTS (SELECT 1 FROM CUSTOMER WHERE ID = @CustomerID)  BEGIN  RAISERROR (N'Клиент с указанным ID не существует.', 16, 1);  END  SELECT \* FROM BOOK\_ORDER WHERE CUSTOMER\_ID = @CustomerID;  END;  CREATE PROCEDURE UPDATE\_ORDER\_STATUS  @OrderID INT,  @NewStatus VARCHAR(20)  AS  BEGIN  IF NOT EXISTS (SELECT 1 FROM BOOK\_ORDER WHERE ID = @OrderID)  BEGIN  RAISERROR (N'Заказ с указанным ID не существует.', 16, 1);  END  UPDATE BOOK\_ORDER SET STATUS = @NewStatus WHERE ID = @OrderID;  END; |

Листинг 3.1 – Скрипт для создания хранимых процедур

1. **Функции в СУБД MS SQL Server**

**Функция GET\_FULL\_AUTHOR\_NAME:**

* Возвращаемый тип: NVARCHAR(150)
* Параметры:
* @AuthorID (INT): Идентификатор автора.
* Назначение: возвращает полное имя автора (имя, фамилия, отчество) по его идентификатору.
* Зависимости: использует таблицу AUTHOR.

**Функция GET\_TOTAL\_ORDERS\_FOR\_CUSTOMER:**

* Возвращаемый тип: INT
* Параметры:
* @CustomerID (INT): Идентификатор клиента.
* Назначение: возвращает общее количество заказов для указанного клиента.
* Зависимости: использует таблицу BOOK\_ORDER.

**Функция GET\_GENRE\_FOR\_BOOK:**

* Возвращаемый тип: NVARCHAR(50)
* Параметры:
* @BookID (INT): Идентификатор книги.
* Назначение: возвращает название жанра книги по её идентификатору.
* Зависимости: использует таблицы BOOK и GENRE.

Скрипт для создания функций приведен в листинге 4.1.

|  |
| --- |
| CREATE FUNCTION GET\_FULL\_AUTHOR\_NAME (@AuthorID INT)  RETURNS NVARCHAR(150)  AS  BEGIN  DECLARE @FullName NVARCHAR(150);  SELECT @FullName = CONCAT(FIRST\_NAME, ' ', LAST\_NAME, ' ', PATRONYMIC)  FROM AUTHOR  WHERE ID = @AuthorID;  IF @FullName IS NULL  BEGIN  SET @FullName = N'Автор не найден';  END  RETURN @FullName;  END;  CREATE FUNCTION GET\_TOTAL\_ORDERS\_FOR\_CUSTOMER (@CustomerID INT)  RETURNS INT  AS  BEGIN  DECLARE @TotalOrders INT;  SELECT @TotalOrders = COUNT(ID)  FROM BOOK\_ORDER  WHERE CUSTOMER\_ID = @CustomerID;  RETURN @TotalOrders;  END;  CREATE FUNCTION GET\_GENRE\_FOR\_BOOK (@BookID INT)  RETURNS NVARCHAR(50)  AS  BEGIN  DECLARE @Genre NVARCHAR(50);  SELECT @Genre = g.NAME  FROM BOOK b  INNER JOIN GENRE g ON b.GENRE\_ID = g.ID  WHERE b.ID = @BookID;  IF @Genre IS NULL  BEGIN  SET @Genre = N'Жанр не указан';  END  RETURN @Genre;  END; |

Листинг 4.1 – Скрипт для создания функций

1. **Список таблиц в СУБД Oracle**

**Таблица AUTHOR**

|  |  |  |  |
| --- | --- | --- | --- |
| Название атрибута | Тип данных | Ограничения | Назначение |
| ID | INT | Primary key, identity | Уникальный идентификатор автора |
| FIRST\_NAME | NVARCHAR(50) | Not null | Имя автора |
| LAST\_NAME | NVARCHAR(50) | Not null | Фамилия автора |
| PATRONYMIC | NVARCHAR(50) | Not null | Отчество автора |
| COUNTRY | NVARCHAR(50) | Not null | Страна происхождения автора |

**Таблица GENRE**

|  |  |  |  |
| --- | --- | --- | --- |
| Название атрибута | Тип данных | Ограничения | Назначение |
| ID | INT | Primary key, identity | Уникальный идентификатор жанра |
| NAME | NVARCHAR(50) | Not null | Название жанра |
| AGE\_RATING | NVARCHAR(10) | Not null | Возрастной рейтинг жанра |

**Таблица BOOK**

|  |  |  |  |
| --- | --- | --- | --- |
| Название атрибута | Тип данных | Ограничения | Назначение |
| ID | INT | Primary key, identity | Уникальный идентификатор книги |
| AUTHOR\_ID | INT | Not null, foreign key | Внешний ключ на таблицу AUTHOR |
| GENRE\_ID | INT | Not null, foreign key | Внешний ключ на таблицу GENRE |
| NAME | NVARCHAR(100) | Not null | Название книги |
| YEAR\_OF\_PUBLISHING | INT |  | Год публикации книги |

**Таблица CUSTOMER**

|  |  |  |  |
| --- | --- | --- | --- |
| Название атрибута | Тип данных | Ограничения | Назначение |
| ID | INT | Primary key, identity | Уникальный идентификатор клиента |
| FIRST\_NAME | NVARCHAR(70) | Not null | Имя заказчика |
| LAST\_NAME | NVARCHAR(70) | Not null | Фамилия заказчика |
| PATRONYMIC | NVARCHAR(70) | Not null | Отчество заказчика |
| PHONE\_NUMBER | VARCHAR(20) | Not null | Номер телефона заказчика |
| EMAIL | VARCHAR(70) | Not null | Электронная почта заказчика |
| TYPE | NVARCHAR(20) | CHECK (TYPE IN (N'Юридическое лицо', N'Физическое лицо')) | Тип заказчика (юридическое или физическое лицо) |

**Таблица BOOK\_CHARACTERISTICS**

|  |  |  |  |
| --- | --- | --- | --- |
| Название атрибута | Тип данных | Ограничения | Назначение |
| ID | INT | Primary key, identity | Уникальный идентификатор характеристики книги |
| COVER\_TYPE | NVARCHAR(50) | Not null | Тип обложки книги |
| PAPER\_TYPE | NVARCHAR(50) | Not null | Тип бумаги |
| BINDING\_TYPE | NVARCHAR(50) | Not null | Тип переплета |
| NUMBER\_OF\_PAGES | INT | Not null | Количество страниц |
| PAPER\_SIZE | NVARCHAR(50) | Not null | Размер бумаги |

**Таблица BOOK\_ORDER**

|  |  |  |  |
| --- | --- | --- | --- |
| Название атрибута | Тип данных | Ограничения | Назначение |
| ID | INT | Primary key, identity | Уникальный идентификатор заказа |
| BOOK\_ID | INT | Not null, foreign key | Внешний ключ на таблицу BOOK |
| CUSTOMER\_ID | INT | Not null, foreign key | Внешний ключ на таблицу CUSTOMER |
| BOOK\_CHARACTERISTICS\_ID | INT | Not null, foreign key | Внешний ключ на таблицу BOOK\_CHARACTERISTICS |
| DATE\_OF\_ORDER | DATE | Not null | Дата заказа книги |
| EDITION | INT | Not null | Тираж книги |
| TOTAL\_PRICE | DECIMAL(10, 2) | Not null | Общая цена заказа |
| STATUS | VARCHAR(20) | CHECK (STATUS IN (N'Создан', N'В работе', N'Готов')) | Статус заказа (создан, в работе, готов) |

1. **Прочие объекты в СУБД Oracle**

**BOOK\_DETAILS:**

* Тип: представление.
* Назначение: предоставляет детальную информацию о книгах, включая данные об авторе и жанре.
* Зависимые объекты: зависит от таблиц BOOK, AUTHOR и GENRE.

**CUSTOMER\_ORDER\_SUMMARY:**

* Тип: представление.
* Назначение: предоставляет сводку о заказах клиентов, включая количество заказов и общую сумму заказов.
* Зависимые объекты: зависит от таблиц CUSTOMER и BOOK\_ORDER.

**ORDER\_DETAILS:**

* Тип: представление.
* Назначение: предоставляет детальную информацию о заказах книг, включая данные о книге, авторе, жанре, характеристиках книги и статусе заказа.
* Зависимые объекты: зависит от таблиц BOOK**\_**ORDER, BOOK, AUTHOR, GENRE и BOOK\_CHARACTERISTICS.

**IDX\_BOOK\_NAME:**

* Тип: индекс.
* Назначение: индексирует столбец NAME в таблице BOOK для повышения производительности при выполнении запросов, которые используют сортировки по этому полю.

**IDX\_AUTHOR\_NAME:**

* Тип: индекс.
* Назначение: индексирует столбцы LAST**\_**NAME, FIRST**\_**NAME и PATRONYMIC в таблице AUTHOR для повышения производительности при выполнении запросов, которые используют эти поля.

**IDX\_ORDER\_DATE:**

* Тип: индекс.
* Назначение: индексирует столбец DATE\_OF\_ORDER в таблице BOOK**\_**ORDER для повышения производительности при выполнении запросов, использующих сортировки по этому полю.

**ORDER\_RECORD:**

* Тип: тип коллекции.
* Назначение: тип, описывающий одну строку таблицы BOOK**\_**ORDER.

**ORDER\_TABLE:**

* Тип: тип коллекции.
* Назначение: тип, описывающий таблицу BOOK**\_**ORDER, состоит из объектов типа ORDER\_RECORD.

Скрипт для создания объектов приведен в листинге 6.1.

|  |
| --- |
| CREATE VIEW BOOK\_DETAILS AS  SELECT b.ID,  b.NAME AS BOOK\_NAME,  a.FIRST\_NAME,  a.LAST\_NAME,  a.PATRONYMIC,  g.NAME AS GENRE\_NAME,  g.AGE\_RATING  FROM BOOK b  INNER JOIN AUTHOR a ON b.AUTHOR\_ID = a.ID  INNER JOIN GENRE g ON b.GENRE\_ID = g.ID;  CREATE VIEW CUSTOMER\_ORDER\_SUMMARY AS  SELECT c.ID,  c.FIRST\_NAME,  c.LAST\_NAME,  c.PATRONYMIC,  COUNT(o.ID) AS TOTAL\_ORDERS,  SUM(o.TOTAL\_PRICE) AS TOTAL\_PRICE  FROM CUSTOMER c  LEFT JOIN BOOK\_ORDER o ON c.ID = o.CUSTOMER\_ID  GROUP BY c.ID, c.FIRST\_NAME, c.LAST\_NAME, c.PATRONYMIC;  CREATE VIEW ORDER\_DETAILS AS  SELECT o.ID , b.NAME AS BOOK\_NAME,  a.FIRST\_NAME,  a.LAST\_NAME,  a.PATRONYMIC,  g.NAME AS GENRE,  bc.COVER\_TYPE, bc.PAPER\_TYPE,  bc.BINDING\_TYPE, bc.NUMBER\_OF\_PAGES,  bc.PAPER\_SIZE,  o.DATE\_OF\_ORDER, o.EDITION,  o.TOTAL\_PRICE, o.STATUS  FROM BOOK\_ORDER o  INNER JOIN BOOK b ON o.BOOK\_ID = b.ID  INNER JOIN AUTHOR a ON b.AUTHOR\_ID = a.ID  INNER JOIN GENRE g ON b.GENRE\_ID = g.ID  INNER JOIN BOOK\_CHARACTERISTICS bc ON o.BOOK\_CHARACTERISTICS\_ID = bc.ID;  CREATE INDEX IDX\_BOOK\_NAME ON BOOK (NAME);  CREATE INDEX IDX\_AUTHOR\_NAME ON AUTHOR (LAST\_NAME, FIRST\_NAME, PATRONYMIC);  CREATE INDEX IDX\_ORDER\_DATE ON BOOK\_ORDER (DATE\_OF\_ORDER); |

Листинг 6.1 – Скрипт для создания представлений и индексов

1. **Хранимые процедуры в СУБД Oracle**

**Хранимая процедура ADD\_BOOK:**

* Параметры:
* Name (NVARCHAR(100)): Название книги.
* AuthorID (INT): Идентификатор автора книги.
* YearOfPublishing (INT): Год публикации книги.
* GenreID (INT): Идентификатор жанра книги.
* Назначение: добавляет новую книгу в базу данных.
* Зависимости: использует таблицу BOOK и проверяет наличие автора и жанра в таблицах AUTHOR и GENRE.

**Хранимая процедура DELETE\_BOOK:**

* Параметры:
* BookID (INT): Идентификатор книги для удаления.
* Назначение: удаляет книгу из базы данных.
* Зависимости: использует таблицу BOOK.

**Хранимая процедура UPDATE\_ORDER\_STATUS:**

* Параметры:
* OrderID (INT): Идентификатор заказа.
* NewStatus (VARCHAR(20)): Новый статус заказа.
* Назначение: обновляет статус заказа.
* Зависимости: использует таблицу BOOK\_ORDER.

Скрипт для создания хранимых процедур приведен в листинге 7.1.

|  |
| --- |
| CREATE OR REPLACE PROCEDURE ADD\_BOOK(  Name IN NVARCHAR2,  AuthorID IN INT,  YearOfPublishing IN INT,  GenreID IN INT  )  AS  author\_count INT;  BEGIN  SELECT COUNT(\*)  INTO author\_count  FROM AUTHOR  WHERE ID = AuthorID;  IF author\_count = 0 THEN  RAISE\_APPLICATION\_ERROR(-20001, 'Автор с указанным ID не существует.');  END IF;  INSERT INTO BOOK (NAME, AUTHOR\_ID, YEAR\_OF\_PUBLISHING, GENRE\_ID)  VALUES (Name, AuthorID, YearOfPublishing, GenreID);  END ADD\_BOOK;  CREATE OR REPLACE PROCEDURE DELETE\_BOOK(  BookID IN INT  )  AS  book\_count INT;  BEGIN  SELECT COUNT(\*)  INTO book\_count  FROM BOOK  WHERE ID = BookID;  IF book\_count = 0 THEN  RAISE\_APPLICATION\_ERROR(-20002, 'Книга с указанным ID не существует.');  END IF;  DELETE  FROM BOOK  WHERE ID = BookID;  END DELETE\_BOOK;  CREATE OR REPLACE PROCEDURE UPDATE\_ORDER\_STATUS(  OrderID IN INT,  NewStatus IN VARCHAR2  )  AS  order\_count INT;  BEGIN  SELECT COUNT(\*)  INTO order\_count  FROM BOOK\_ORDER  WHERE ID = OrderID;  IF order\_count = 0 THEN  RAISE\_APPLICATION\_ERROR(-20004, 'Заказ с указанным ID не существует.');  END IF;  UPDATE BOOK\_ORDER  SET STATUS = NewStatus  WHERE ID = OrderID;  END UPDATE\_ORDER\_STATUS; |

Листинг 7.1 – Скрипт для создания хранимых процедур

1. **Функции в СУБД Oracle**

**Функция GET\_FULL\_AUTHOR\_NAME:**

* Возвращаемый тип: NVARCHAR(150)
* Параметры:
* AuthorID (INT): Идентификатор автора.
* Назначение: возвращает полное имя автора (имя, фамилия, отчество) по его идентификатору.
* Зависимости: использует таблицу AUTHOR.

**Функция GET\_TOTAL\_ORDERS\_FOR\_CUSTOMER:**

* Возвращаемый тип: INT
* Параметры:
* CustomerID (INT): Идентификатор клиента.
* Назначение: возвращает общее количество заказов для указанного клиента.
* Зависимости: использует таблицу BOOK\_ORDER.

**Функция GET\_GENRE\_FOR\_BOOK:**

* Возвращаемый тип: NVARCHAR(50)
* Параметры:
* BookID (INT): Идентификатор книги.
* Назначение: возвращает название жанра книги по её идентификатору.
* Зависимости: использует таблицы BOOK и GENRE.

**Функция GET\_CUSTOMER\_ORDERS:**

* Возвращаемый тип: ORDER\_TABLE
* Параметры:
* CustomerID (INT): Идентификатор клиента.
* Назначение: получает заказы клиента по его идентификатору.
* Зависимости: использует таблицу CUSTOMER и BOOK\_ORDER.

Скрипт для создания функций приведен в листинге 8.1.

|  |
| --- |
| CREATE OR REPLACE FUNCTION GET\_CUSTOMER\_ORDERS(  CustomerID IN INT  )  RETURN ORDER\_TABLE  AS  customer\_count INT;  v\_result ORDER\_TABLE := ORDER\_TABLE();  BEGIN  SELECT COUNT(\*)  INTO customer\_count  FROM CUSTOMER  WHERE ID = CustomerID;  IF customer\_count = 0 THEN  RAISE\_APPLICATION\_ERROR(-20003, 'Клиент с указанным ID не существует.');  END IF;  SELECT ORDER\_RECORD(ID, BOOK\_ID, CUSTOMER\_ID, BOOK\_CHARACTERISTICS\_ID, DATE\_OF\_ORDER, EDITION, TOTAL\_PRICE, STATUS)  BULK COLLECT  INTO v\_result  FROM BOOK\_ORDER  WHERE CUSTOMER\_ID = CustomerID;  RETURN v\_result;  END GET\_CUSTOMER\_ORDERS;  CREATE OR REPLACE FUNCTION GET\_FULL\_AUTHOR\_NAME (  AuthorID IN INT  )  RETURN NVARCHAR2  AS  FullName NVARCHAR2(150);  BEGIN  SELECT FIRST\_NAME || ' ' || LAST\_NAME || ' ' || PATRONYMIC  INTO FullName  FROM AUTHOR  WHERE ID = AuthorID;  IF FullName IS NULL THEN  RAISE\_APPLICATION\_ERROR(-20001, 'Автор с указанным ID не найден.');  END IF;  RETURN FullName;  END GET\_FULL\_AUTHOR\_NAME;  CREATE OR REPLACE FUNCTION GET\_TOTAL\_ORDERS\_FOR\_CUSTOMER (  CustomerID IN INT  )  RETURN INT  AS  TotalOrders INT;  BEGIN  SELECT COUNT(ID)  INTO TotalOrders  FROM BOOK\_ORDER  WHERE CUSTOMER\_ID = CustomerID;  RETURN TotalOrders;  END GET\_TOTAL\_ORDERS\_FOR\_CUSTOMER;  CREATE OR REPLACE FUNCTION GET\_GENRE\_FOR\_BOOK (  BookID IN INT  )  RETURN NVARCHAR2  AS  Genre NVARCHAR2(50);  BEGIN  SELECT g.NAME  INTO Genre  FROM BOOK b  INNER JOIN GENRE g ON b.GENRE\_ID = g.ID  WHERE b.ID = BookID;  IF Genre IS NULL THEN  RAISE\_APPLICATION\_ERROR(-20002, 'Жанр для указанной книги не найден.');  END IF;  RETURN Genre;  END GET\_GENRE\_FOR\_BOOK; |

Листинг 8.1 – Скрипт для создания функций