

assignment

by Carys Rees

Submission date: 07-Jan-2025 04:18PM (UTC+0000)

Submission ID: 247523326

File name: 23703_Carys_Rees_assignment_621315_1555023017.docx (662.82K)

Word count: 1809

Character count: 11139



Prifysgol Cymru
Y Drindod Dewi Sant
University of Wales
Trinity Saint David

Databases and Application
Development (ACCA5026)

Adventure Works Cycles' Database
Design & Implementation

Carys Rees 2003859

Contents

Introduction	3
Database Design.....	4
Entity Relationship Diagram (ERD).....	4
Database Schema	5
3 rd Normal Form (3NF)	5
Table Structures and Relationships	6
Database Implementation	6
Table Creation	6
Referential Integrity Constraints	6
Check Constraints	7
Populating the Database with Sample Data	7
Transactional Scenario Demonstration	7
Performance Tuning Enhancement.....	8
Identifying Key Queries and Indexing Strategy	8
Screenshots and Evidence.....	8
Conclusion	9
References	10

Introduction

AdventureWorks Cycles is a well-established bicycle manufacturing company seeking to expand its business by creating an advanced online marketplace. This new platform aims to offer a wide range of bicycles and accessories, allowing customers to browse, customise, and purchase products online. To support this transition, AdventureWorks Cycles needs a robust and efficient database infrastructure to manage the core aspects of its operations, including product listings, orders, customer management, and manufacturer information.

The goal of this assignment is to design and implement a comprehensive database solution that supports the business processes of AdventureWorks Cycles. This involves creating a database schema, implementing referential integrity through foreign key constraints, defining data validation rules with check constraints, and populating the database with sample records. The solution also includes handling transactional operations through rollback and commit functionality, and optimising performance using indexing strategies to improve query efficiency.

This report outlines the database design and implementation process for AdventureWorks Cycles, demonstrating the creation of necessary tables, relationships, constraints, and testing of various database operations. The implementation will ensure that the database maintains data integrity, supports efficient transactions, and is scalable as the online marketplace grows.

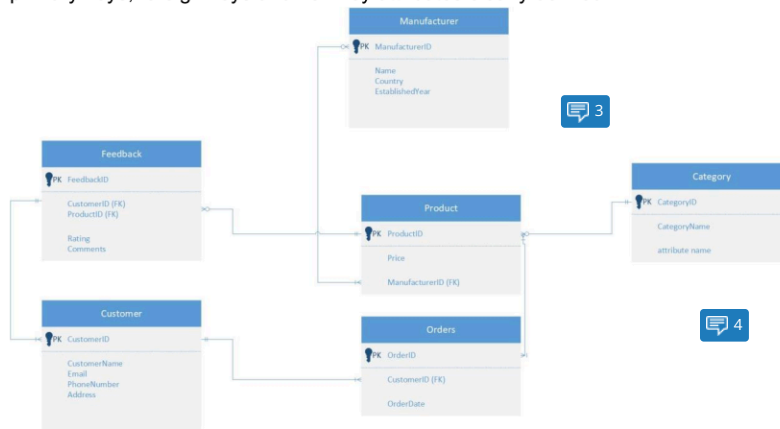


Database Design

Entity Relationship Diagram (ERD)

The entity relationship diagram (ERD) for the AdventureWorks Cycles database represents the entities in the business system and the relationships between them.

[1]. The ERD consists of several entities such as Manufacturer, Product, Order, Customer, Category and Feedback. Each entity is represented as a table, with primary keys, foreign keys and non-key attributes clearly defined.



The relationships between entities are outlined in the ERD:

- **Manufacturer:** Each manufacturer produces one or more products, forming a one-to-any relationship between Manufacturer and Product.
- **Product:** Products are linked to specific manufacturers via a foreign key reference. The product table also holds product details such as price, availability, and product name.
- **Order:** Customers place orders, and each order can contain multiple products. Therefore, the order table is connected to the customer table through a foreign key, and the relationship between Order and Product is many-to-many.
- **Customer:** Each customer can place multiple orders. The customer table contains customer-specific information such as name, email, and address.
- **Feedback:** Each customer is also able to leave feedback, which will link up to the customer table and product table, also allowing ratings and comments.

In the ERD, primary keys uniquely identify each record within a table, while foreign keys ensure that relationships between tables are preserved. [2]. Non-key attributes provide additional details about the entities they represent.

Database Schema

The database schema for AdventureWorks Cycles includes the creation of tables that represent key business entities such as Manufacturer, Product, Customer, and Order. The schema is designed to ensure the integrity of the data and to support the core functionalities required for managing products, orders, and customers.

Each table in the schema is designed to contain specific attributes. For example, the Product table holds details such as product name, price, and associated manufacturer. The Customer table contains customer details like name and contact information. The Order table manages the orders placed by customers, including order data and associated products.

The schema includes a combination of primary keys, foreign keys, and non-key attributes that serve to link the tables and establish the relationships between them. [3].

3rd Normal Form (3NF)

The database schema has been normalised to the third normal form (3NF) to ensure data integrity, reduce redundancy, and improve the maintainability of the database. [4]. The normalisation process involves the following steps:

1. **First Normal Form (1NF):** Guarantees that every attribute has atomic values, or that there is only one value in every field.
2. **Second Normal Form (2NF):** By guaranteeing that  non-key property is completely functionally reliant on the entire primary key, partial dependencies are eliminated.
3. **Third Normal Form (3NF):** By guaranteeing that non-key characteristics rely solely on the primary key and not on other non-key attributes, transitive dependencies are removed.

By adhering to 3NF, the database structure avoids anomalies such as redundant data and ensures that the database can handle various updates, deletions, and insertions without violating integrity.

Table Structures and Relationships

- **Manufacturer Table:** Contains attributes such as ManufacturerID (primary key) and ManufacturerName. This table serves as the reference for the products manufactured by different companies.
- **Product Table:** Contains ProductID (primary key), ProductName, Price, and ManufacturerID (foreign key), linking to the Manufacturer table. The Price attribute is constrained to ensure that only positive values are allowed.
- **Order Table:** Contains OrderID (primary key), CustomerID (foreign key), and OrderDate. This table records the details of customer orders and is linked to the Customer table by the CustomerID.
- **Customer Table:** Contains CustomerID (primary key), CustomerName, and other relevant contact details. The table links customers to their orders via the Order table.
- **Feedback Table:** Contains FeedbackID (primary key), CustomerID (foreign key), ProductID (foreign key), rating and comments. This table links feedback to their orders.
- **Order Details:** Contains OrderDetailID (primary key), OrderID (foreign key), ProductID (foreign key), and quantity. This table allows the customers to link to the order table.

These relationships ensure the accurate tracking of product details, customer orders, and other critical information within the database.

Database Implementation

Table Creation



The implementation of the database schema involves the creation of tables based on the ERD and the schema design. Each table is created using SQL statements that define its structure, including primary keys, foreign keys, and constraints. These tables serve as the foundation for managing data related to products, customers, and orders.

Referential Integrity Constraints

Referential Integrity is maintained using foreign keys, ensuring that relationships between tables are preserved. For example, the Product table includes a foreign key reference to the Manufacturer table, which ensures that each product is associated with an existing manufacturer. Likewise, the Order table references the Customer table, ensuring that each order is linked to a valid customer.

Foreign keys prevent the insertion of invalid or inconsistent data by ensuring that only valid references are made between tables. If a customer does not exist, an order cannot be placed for that customer.

Check Constraints

Check constraints are used to enforce data integrity rules at the database level. Two check constraints were implemented in the Product and Order tables:

Price Check: A check constraint on the Price attribute in the Product table ensures that product prices are always greater than zero.

Order Date Check: A check constraint on the OrderData attribute in the Order table ensures that orders cannot be placed for future dates.  7

These constraints ensure that only valid data is entered into the database, preventing errors or inconsistencies in the dataset.

Populating the Database with Sample Data

Sample data was inserted into the tables to test the functionality of the primary key, foreign key, and check constraints. This data includes information about manufacturers, products, customers, and orders, all of which are necessary for verifying that the database design works as expected. Insert statements were carefully crafted to ensure the integrity of the data, with no violations of constraints.

Transactional Scenario Demonstration

A key feature of the AdventureWorks Cycles database is the ability to perform transactional operations that involve multiple SQL statements. These operations ensure that changes to the database are consistent, and the database maintains its integrity even in the case of errors or unexpected events.

In this section, a transactional scenario was created to demonstrate the use of rollback and commit operations. The scenario involves a customer attempting to place an order for a product, where the system checks for the product's availability. If the product is available, the transaction is committed, and the order is processed. If the product is out of stock, the transaction is rolled back, and the order is not placed.

The rationale behind this approach is to ensure that only valid transactions are committed, maintaining data consistency and preventing errors or incomplete operations from being saved to the database.

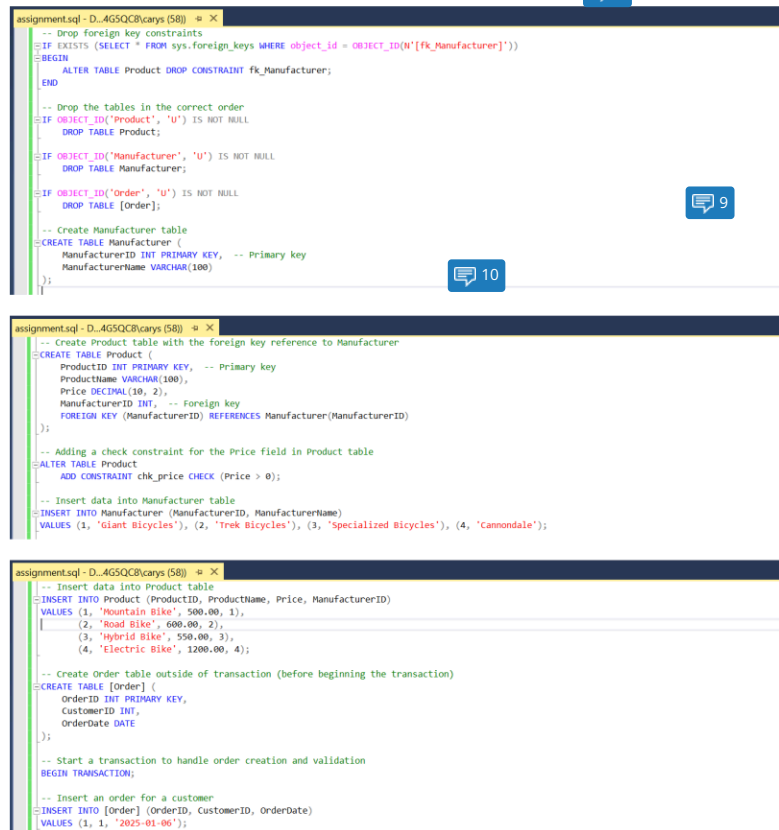
Performance Tuning Enhancement

Identifying Key Queries and Indexing Strategy

Performance optimisation is a critical aspect of database design, especially as the size and complexity of the database grows. To enhance query performance, secondary indexes were created on columns that are frequently used in queries, such as CustomerID in the Order table and ProductID in the Product table.

These indexes improve the performance of SELECT queries by allowing the database engine to quickly locate records based on indexed columns, reducing the time spent scanning large tables. Indexing these columns is particularly useful for frequently executed queries that involve filtering by customer or product.

Screenshots and Evidence



The screenshots show a sequence of SQL commands in a text editor window titled 'assignment.sql - D:\4G5QCB\carys (58)'. The first screenshot shows the removal of a foreign key constraint and the dropping of the Product and Manufacturer tables. The second screenshot shows the creation of the Manufacturer table, the Product table with a foreign key to Manufacturer, and the addition of a check constraint for the Price field. The third screenshot shows the insertion of data into the Manufacturer table, the insertion of data into the Product table, and the creation of the Order table. Finally, a transaction is started, and an order is inserted for a customer.

```
-- Drop foreign key constraints
IF EXISTS (SELECT * FROM sys.foreign_keys WHERE object_id = OBJECT_ID(N'[fk_Manufacturer]'))
BEGIN
    ALTER TABLE Product DROP CONSTRAINT fk_Manufacturer;
END

-- Drop the tables in the correct order
IF OBJECT_ID('Product', 'U') IS NOT NULL
    DROP TABLE Product;

IF OBJECT_ID('Manufacturer', 'U') IS NOT NULL
    DROP TABLE Manufacturer;

IF OBJECT_ID('Order', 'U') IS NOT NULL
    DROP TABLE [Order];

-- Create Manufacturer table
CREATE TABLE Manufacturer (
    ManufacturerID INT PRIMARY KEY, -- Primary key
    ManufacturerName VARCHAR(100)
);

-- Create Product table with the foreign key reference to Manufacturer
CREATE TABLE Product (
    ProductID INT PRIMARY KEY, -- Primary key
    ProductName VARCHAR(100),
    Price DECIMAL(10, 2),
    ManufacturerID INT, -- Foreign key
    FOREIGN KEY (ManufacturerID) REFERENCES Manufacturer(ManufacturerID)
);

-- Adding a check constraint for the Price field in Product table
ALTER TABLE Product
    ADD CONSTRAINT chk_price CHECK (Price > 0);

-- Insert data into Manufacturer table
INSERT INTO Manufacturer (ManufacturerID, ManufacturerName)
VALUES (1, 'Giant Bicycles'), (2, 'Trek Bicycles'), (3, 'Specialized Bicycles'), (4, 'Cannondale');

-- Insert data into Product table
INSERT INTO Product (ProductID, ProductName, Price, ManufacturerID)
VALUES (1, 'Mountain Bike', 500.00, 1),
       (2, 'Road Bike', 600.00, 2),
       (3, 'Hybrid Bike', 550.00, 3),
       (4, 'Electric Bike', 1200.00, 4);

-- Create Order table outside of transaction (before beginning the transaction)
CREATE TABLE [Order] (
    OrderID INT PRIMARY KEY,
    CustomerID INT,
    OrderDate DATE
);

-- Start a transaction to handle order creation and validation
BEGIN TRANSACTION;

-- Insert an order for a customer
INSERT INTO [Order] (OrderID, CustomerID, OrderDate)
VALUES (1, 1, '2025-01-06');
```

```

assignment.sql - D:\G5QCB\carys (58)
-- Simulate a problem (e.g. Product is out of stock)
-- IF (SELECT COUNT(*) FROM Product WHERE ProductID = 999) = 0
-- BEGIN
--     ROLLBACK; -- Rollback if product does not exist
--     PRINT 'Transaction rolled back';
-- END
-- ELSE
-- BEGIN
--     COMMIT; -- Commit if product exists
--     PRINT 'Transaction committed';
-- END
-- Create indexes for better performance after the table creation
CREATE INDEX idx_customer_id ON [order](CustomerID);
CREATE INDEX idx_product_id ON Product(ProductID);

```

The above screenshots show the SQL code executed for AdventureWorks Cycles.

Conclusion

The database design and implementation for AdventureWorks Cycles demonstrate a comprehensive approach to managing the company's operations efficiently. The structured database schema, developed using an Entity Relationship Diagram, ensures that all critical entities such as Manufacturer, Product, Customer, and Order and properly represented. Adhering to the principles of normalisation up to the third normal form has minimised data redundancy, ensured data integrity, and facilitated efficient data management.

Key features of the implementation include the use of referential integrity constraints to maintain relationships between entities and the addition of check constraints to enforce business rules, such as ensuring positive product prices and valid order dates. Realistic sample data has been used to validate these constraints, confirming the functionality of primary and foreign keys and ensuring that the database meets its intended purpose.

The demonstration of transactional scenarios highlights the importance of robust error handling and consistency in database operations, using rollback and commit mechanisms to manage complex transactions. Additionally, the performance tuning enhancements, such as indexing frequently queried columns, ensure that the database is optimised for speed and scalability, supporting future growth and operational demands.

This project has provided a solid foundation for AdventureWorks Cycles to manage its data effectively while maintaining high standards of accuracy, consistency, and efficiency. The design and implementation process also sets the stage for future enhancements, ensuring the database can adapt to the evolving needs of the business.

References

1. "Conceptual Database Model. Entity Relationship Diagram (ERD)." Available: https://dbs.academy.lv/lection/dbs_LS04EN_erd.pdf
2. L. Jiang and F. Naumann, "Holistic primary key and foreign key detection," *Journal of Intelligent Information Systems*, vol. 54, no. 3, pp. 439–461, Jun. 2019, doi: <https://doi.org/10.1007/s10844-019-00562-z>.
3. "Entity Relationship Diagram -ER Diagram in DBMS." Available: https://www.du.ac.in/du/uploads/departments/Operational%20Research/24042020_E-R%20Model.pdf
4. T. W. Ling, "A Normal Form For Entity-Relationship Diagrams.," *ResearchGate*, pp. 24–35, 1985, Accessed: Jan. 07, 2025. [Online]. Available: https://www.researchgate.net/publication/221269712_A_Normal_Form_For_Entity-Relationship_Diagrams

assignment

GRADEMARK REPORT

FINAL GRADE

GENERAL COMMENTS

45/100

PAGE 1

PAGE 2



Comment 1 | Transferable Skills

clear ToC however numbering helps readers quickly locate sections and sub-sections based on their position in the document, creating a logical structure

PAGE 3



Comment 2 | Transferable Skills

The purpose of the report is well-defined, explaining that it focuses on database design, implementation

PAGE 4



Comment 3 | Knowledge&Understanding Skills

entities may be missing important attributes.



Comment 4 | Knowledge&Understanding Skills

diagram does not differentiate between strong and weak entities

PAGE 5



Comment 5 | Knowledge&Understanding Skills

explanation of how normalization was applied is missing

PAGE 6



Comment 6 | Cognitive &Intellectual Skills

evidence of table creation missing

PAGE 7



Comment 7 | Practical Application Skills



Comment 8 | Cognitive &Intellectual Skills

It is important to number all figures and tables sequentially and ensure that labels are consistent throughout the report



Comment 9 | Cognitive &Intellectual Skills

all the required screenshots are not included



Comment 10 | Practical Application Skills

no mention of which specific database management system has been used.



Comment 11 | Transferable Skills

conclusion provides a clear sense of closure to the report



Comment 12 | Transferable Skills

Reflect on technical challenges and how they were solved, offering more depth in the conclusions drawn from the work



Comment 13 | Literature Skills

literature review cites few authoritative sources

GRADING FORM: DAD_2024

CARYS REES

45

LITERATURE SKILLS



3

KNOWLEDGE&UNDERSTANDING SKILLS



8

COGNITIVE &INTELLECTUAL SKILLS



8.5

PRACTICAL APPLICATION SKILLS



14

TRANSFERABLE SKILLS



11.5