SQL PROJECTS ON

FREE COPY

BANKING & FINANCE

(WITH SOLUTIONS)

Eyowhite

SQL PROJECTS ON

BANKING & FINANCE

Dr. Eyo Eyo

Website: *www.eyowhite.com*

Twitter (X): *twitter.com/eyowhite3*

www.eyowhite.com

Database Diagram

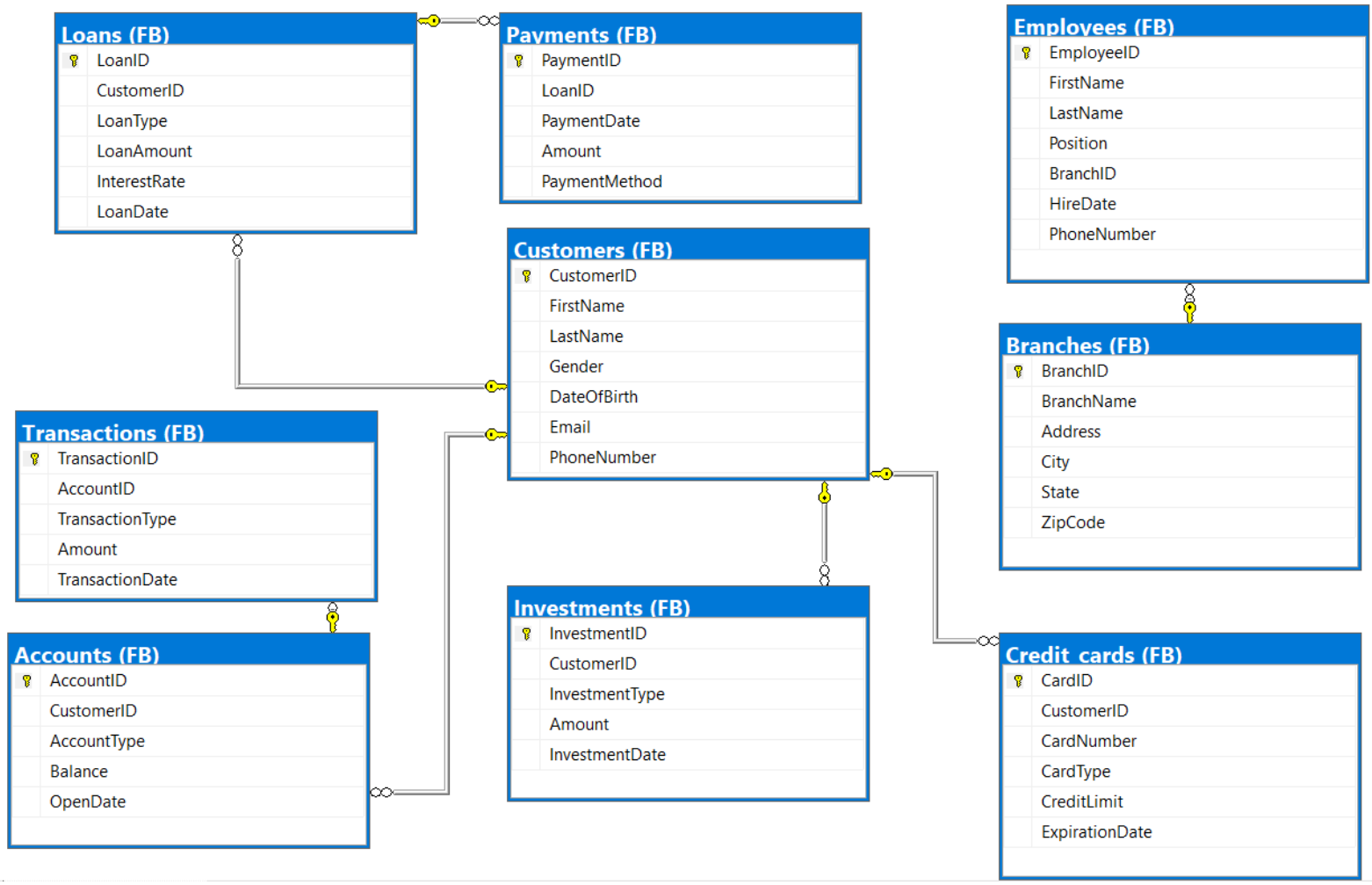


Table of Contents

About the Author

Dr. Eyo Eyo (PhD) is an IT Business Analyst and formerly a University Lecturer. He is an accomplished researcher in the fields of machine learning, data analysis, business analysis, and engineering.

Eyo holds a Doctor of Philosophy (PhD) degree, which stands as a testament to his dedication to advancing knowledge in data science and engineering. His academic journey has been marked by a relentless pursuit of excellence, resulting in numerous scholarly achievements and contributions that have enriched his discipline.

Eyo's commitment to data analytics and artificial intelligence goes beyond the classroom, as he continually seeks innovative ways to engage learners and foster a deeper understanding of complex subjects.

In the realm of research, Eyo has made significant strides. His work in machine learning and data analysis has led to groundbreaking insights and practical applications, contributing to the advancement of knowledge through numerous published works.

As you delve into the pages of Eyo's book, "SQL Projects on Banking and Finance", you will discover the depth of his expertise and the invaluable insights he brings to the world of Structured Query Language and data analysis in general.

Important information

Welcome to "**SQL Projects on Banking and Finance**". This book is designed to be your comprehensive guide to mastering the art of SQL using practical business scenarios. Whether you're a beginner looking to build a strong foundation or an experienced practitioner aiming to sharpen your skills.

SQL (Structured Query Language) is the backbone of managing and manipulating data in modern databases, making it an essential skill for anyone working with data-driven applications.

This book covers the commands used in Data Query language (DQL) in most parts. Other categories of SQL commands namely, Data Manipulation Language (DML), Data Definition Language (DDL) and Data Control Language (DCL) will be covered in a separate book.

Solutions to the problems in this book are given at the end of this book (before the Appendix).

NOTE: *The commands employed in the "sample" solutions primarily align with SQL Server conventions, although suggestions are provided on how the syntax might be adapted to suit other frequently used SQL platforms such as MySQL, Oracle, PostgreSQL, etc.*

Database used in this book.

The database used in this book is based solely on Banking and Finance as the title suggests. Access to the database should have already been acquired when the book was downloaded. The Database Diagram given above (after the copyright page) can help in understanding the relationships between the tables.

SQL Server installation and database restoration

In the Appendix, you'll find comprehensive instructions for installing SQL Server 2019 (or later) on Windows 10 and later versions, as well as the step-by-step process on how to restore a database using the Microsoft AdventureWorks database as an example.

Who This Book Is For

This **"SQL Projects on Banking and Finance"** assumes some fundamental knowledge of SQL at the least. Nevertheless, it is intended for a wide audience, including beginners, students, educators and database professionals and anyone who uses SQL in their everyday lives and business.

Feedback

We value your feedback and suggestions. If you have any comments, questions, or ideas for improvement, please don't hesitate to reach out on any of the following platforms:

- Twitter: twitter.com/Eyowhite3
- Website: eyowhite.com/contact

Your input will help us enhance future editions of this book.

Thank you for choosing " SQL Projects on Banking and Finance ". We hope this book serves as a valuable resource in your journey to become a proficient SQL practitioner.

Happy querying!

[Eyo Eyo]

Business Scenarios

1. Business Scenario Q1

Customer Account Balances Overview

The bank management wants to have a comprehensive view of all customers along with their account details and current balances. This information is crucial for identifying high-value customers, understanding the distribution of account balances, and planning targeted marketing campaigns.

2. Business Scenario Q2

High-Value Customers Identification

The bank management wants to identify all customers who have a balance greater than $5,000 in their accounts. This information is critical for understanding the high-value customer segment, offering them tailored financial products, and providing them with premium customer services.

3. Business Scenario Q3

Transactions in 2022

The bank management wants to analyse all transactions made in the year 2022 to understand customer behaviour, transaction volumes, and financial flows during that period. This analysis will help in identifying trends, detecting anomalies, and planning future strategies.

4. Business Scenario Q4

Monthly Deposit Summary

The bank management wants to calculate the total amount deposited in all accounts for the month of May 2022. This information is essential for monitoring cash inflows, assessing the bank's liquidity position, and planning for future financial needs.

5. Business Scenario Q5

Customer Loan Details

The bank management wants to retrieve the details of all loans taken by a customer with ID “C0768”. This information is crucial for understanding the customer's borrowing behaviour, managing their credit risk, and providing them with tailored loan products.

6. Business Scenario Q6

Branch-Specific Employee List

The bank management wants to retrieve a list of all employees working in a branch having an ID “B0851”. This information is useful for branch managers to understand their team composition, manage human resources effectively, and plan for staffing needs.

7. Business Scenario Q7

Total Credit Cards Issued

The bank management wants to determine the total number of credit cards issued by the bank. This information is important for understanding the bank's reach in the credit card market, evaluating the success of their credit card products, and planning future marketing campaigns.

8. Business Scenario Q8

Average Interest Rate for Loans

The bank management wants to calculate the average interest rate for all loans. This information is essential for assessing the overall cost of borrowing for customers, comparing it with industry benchmarks, and making decisions about future loan product offerings and interest rate adjustments.

9. Business Scenario Q9

Active Customers in 2020

The bank management wants to identify and retrieve the details of all customers who have made at least one transaction in the year 2020. This information is valuable for understanding customer activity, identifying engaged customers, and planning targeted marketing and customer retention strategies.

10. Business Scenario Q10

Inactive Accounts Between 2019 and 2023

The bank management wants to identify all accounts that have had no transactions between the years 2019 and 2023. This information is important for understanding long-term account inactivity, identifying dormant accounts, and planning strategies to reactivate these accounts.

11. Business Scenario Q11

Total Loan Payments in 2015

The bank management wants to calculate the total amount of payments made towards loans in the year 2015. This information is essential for understanding the cash flow related to loan repayments, assessing the bank's financial performance for that year, and making strategic decisions based on loan repayment trends.

12. Business Scenario Q12

Customer Investments in Mutual Funds

The bank management wants to retrieve the details of all investments made by customers in mutual funds. This information is valuable for understanding the investment preferences of customers, assessing the performance of mutual fund products, and planning targeted investment offerings.

13. Business Scenario Q13

Transaction Count by Account Type

The bank management wants to find the total number of transactions for each account type (Checking, Savings, Credit). This information is important for understanding the transaction activity across different types of accounts, identifying popular account types, and making strategic decisions related to product offerings and customer engagement.

14. Business Scenario Q14

Employee Count by Branch

The bank management wants to list all branches along with the number of employees working in each branch. This information is essential for understanding branch staffing levels, identifying branches that may need additional staffing, and optimising human resource allocation.

15. Business Scenario Q15

Total Outstanding Loan Amount by Customer

The bank management wants to calculate the total outstanding loan amount for each customer. This information is crucial for assessing individual customer debt levels, managing credit risk, and making informed decisions about loan approvals and customer credit limits.

16. Business Scenario Q16

Customers with Multiple Account Types

The bank management wants to retrieve the details of all customers who have more than one type of account. This information is important for understanding customer engagement, identifying cross-selling opportunities, and analysing the diversity of customer portfolios.

17. Business Scenario Q17

Total Number of Loans Approved in 2017

The bank management wants to find the total number of loans approved in the year 2017. This information is essential for assessing the bank's lending activity for that year, understanding market demand, and planning future loan offerings and strategies.

18. Business Scenario Q18

Average Balance of Savings Accounts

The bank management wants to calculate the average balance of all savings accounts. This information is important for understanding the typical balance held by customers in savings accounts, assessing the bank's liquidity, and making informed decisions about interest rates and savings account products.

19. Business Scenario Q19

Customers with Stock Investments

The bank management wants to retrieve the details of all customers who have investments in stocks. This information is valuable for understanding customer investment behaviour, identifying customers interested in equity markets, and planning targeted marketing campaigns for stock-related financial products.

20. Business Scenario Q20

Total Interest Earned on Loans in 2012

The bank management wants to calculate the total interest earned on all loans in the year 2012. This information is crucial for understanding the revenue generated from

loan interest during that period, evaluating the profitability of the bank's lending activities, and making informed financial planning and strategic decisions.

21. Business Scenario Q21

Total Number of Deposits in a Specific Branch

The bank management wants to calculate the total number of deposit transactions made in a branch with ID “B0036”. This information is essential for understanding the deposit activity within a branch, assessing branch performance, and planning resource allocation.

22. Business Scenario Q22

Employees Hired in 2018

The bank management wants to retrieve the details of all employees who were hired in the year 2018. This information is important for understanding hiring trends, analysing employee retention, and planning future hiring strategies.

23.Business Scenario Q23

Total Amount of Investments Made by All Customers

The bank management wants to calculate the total amount of investments made by all customers. This information is crucial for understanding the overall investment activity, evaluating the bank's investment product performance, and making strategic decisions regarding investment offerings.

24. Business Scenario Q24

Customers with Multiple Loans

The bank management wants to retrieve the details of all customers who have more than one loan. This information is important for understanding customer borrowing behaviour, identifying high-risk customers, and providing targeted financial services.

25. Business Scenario Q25

Accounts with Low Balances

The bank management wants to list all accounts that have a balance less than $500. This information is important for identifying accounts that may require attention, such as those at risk of becoming inactive or needing additional financial products and services to encourage higher balances.

26. Business Scenario Q26

Total Amount of Withdrawals in January 2023

The bank management wants to calculate the total amount of withdrawals made in January 2023. This information is essential for understanding cash outflows, assessing liquidity needs, and planning for financial management and customer service strategies.

27.Business Scenario Q27

Customers Making Payments Using Bank Transfers

The bank management wants to retrieve the details of all customers who have made payments using bank transfers. This information is important for understanding

customer payment preferences, identifying trends in payment methods, and planning targeted services and promotions for bank transfer users.

28. Business Scenario Q28

Total Number of Credit Card Transactions in 2022

The bank management wants to find the total number of credit card transactions made in the year 2022. This information is important for understanding the usage and popularity of credit cards among customers, assessing transaction volumes, and planning marketing strategies for credit card products.

29. Business Scenario Q29

Average Credit Limit of Credit Cards

The bank management wants to calculate the average credit limit of all credit cards. This information is essential for understanding the distribution of credit limits among customers, assessing the bank's credit exposure, and making informed decisions about credit card policies and offerings.

30. Business Scenario Q30

Customers with Bond Investments

The bank management wants to retrieve the details of all customers who have investments in bonds. This information is valuable for understanding customer investment preferences, identifying potential opportunities for targeted marketing of bond-related financial products, and analysing the popularity of bonds among customers.

31. Business Scenario Q31

Total Number of Loans Approved by Loan Type

The bank management wants to calculate the total number of loans approved for each loan type (Personal, Mortgage, Auto, Student). This information is crucial for understanding the distribution of loan approvals across different types, evaluating the demand for various loan products, and making informed decisions about future loan offerings.

32.Business Scenario Q32

List of Employees Working as Loan Officers

The bank management wants to list all employees who work as loan officers. This information is essential for understanding the workforce composition, managing human resources, and planning targeted training and development programs for loan officers.

33.Business Scenario Q33

Total Number of Accounts Opened in 2014

The bank management wants to find the total number of accounts that were opened in the year 2014. This information is important for understanding the growth in the customer base during that year, evaluating the success of marketing campaigns, and making informed decisions about future strategies to attract new customers.

34. Business Scenario Q34

Average Transaction Amount by Transaction Type

The bank management wants to calculate the average transaction amount for each transaction type. This information is essential for understanding customer behaviour, identifying transaction trends, and making informed decisions about fee structures and service offerings.

ADVANCED SCENARIOS

35.Business Scenario Q35

Identify High-Value Customers

The bank management wants to identify high-value customers, defined as the top 10% of customers based on their total account balances and investments. This information is crucial for targeted marketing, offering premium services, and personalised financial products to these valuable customers.

36. Business Scenario Q36

Customer Segmentation

The bank management wants to categorise customers into segments (e.g., low, medium, high value) based on their account balances, transaction frequency, and investment amounts. This information is crucial for targeted marketing, personalised service offerings, and enhancing customer satisfaction.

37.Business Scenario Q37

Account Activity Summary

The bank management wants to retrieve a summary of account activity for each customer. This summary should include the total deposits, total withdrawals, and the current balance for each customer's accounts. This information is crucial for understanding customer behaviour, monitoring account health, and providing personalised financial advice.

38.Business Scenario Q38

Customer Investment Profile

The bank management wants to list customers along with their total investment amounts and the types of investments they hold. This information is crucial for understanding customer investment behaviour, identifying potential high-value clients, and tailoring investment products to meet customer needs.

39. Business Scenario Q39

Monthly Transaction Volume

The bank management wants to calculate the total transaction volume for each month from 2011 to 2023, broken down by transaction type. This information is essential for understanding transaction patterns, planning for resource allocation, and identifying peak transaction periods.

40. Business Scenario Q40

Loan Repayment Status

The bank management wants to list all loans along with their repayment status. This should include the total amount repaid and the outstanding balance for each loan. This information is crucial for monitoring loan performance, identifying potential defaults, and managing credit risk.

41. Business Scenario Q41

Analyse Customer Investment Trends

The bank management wants to analyse customer investment trends, Year-over-Year Growth in Investments: This includes understanding how customers are investing over time, identifying popular investment types, and tracking the total investment amounts. This information is crucial for developing investment products, marketing strategies, and providing personalised investment advice.

42. Business Scenario Q42

Credit Card Expiry Notification

The bank management wants to identify credit cards that are set to expire in the next three months (from January 2022) and list their holders. This information is crucial for customer service, allowing the bank to notify customers about the impending expiration of their credit cards and facilitate timely renewals or replacements.

43. Business Scenario Q43

Customer Investment Portfolio Analysis

The bank management wants to analyse the investment portfolios of their customers. This involves identifying the types of investments each customer has, the total amount invested by each customer, and the distribution of investment types. This information is crucial for understanding customer investment behaviour, tailoring investment products, and providing personalised financial advice.

44. Business Scenario Q44

The bank management wants to analyse the performance of each branch. This includes calculating the total number of accounts, total loan amounts, total transaction amounts, and total number of employees for each branch. This information is crucial for evaluating branch performance, identifying high-performing branches, and making informed decisions about resource allocation and branch management.

SOLUTIONS

```sql
--Q1.
SELECT
    c.CustomerID,
    c.FirstName,
    c.LastName,
    c.Gender,
    c.DateOfBirth,
    c.Email,
    c.PhoneNumber,
    a.AccountID,
    a.AccountType,
    a.Balance,
    a.OpenDate
FROM
    FB.Customers c
JOIN
    FB.Accounts a
ON
    c.CustomerID = a.CustomerID;


--Q2.
SELECT
    c.CustomerID,
    c.FirstName,
    c.LastName,
    c.Gender,
    c.DateOfBirth,
    c.Email,
    c.PhoneNumber,
    a.AccountID,
    a.AccountType,
    a.Balance,
    a.OpenDate
FROM
    FB.Customers c
JOIN
    FB.Accounts a
ON
    c.CustomerID = a.CustomerID
WHERE
    a.Balance > 5000;

--Q3.
SELECT
    t.TransactionID,
    t.AccountID,
    t.TransactionType,
    t.Amount,
    t.TransactionDate
FROM
    FB.Transactions t
WHERE
```

SOLUTIONS

```sql
    t.TransactionDate BETWEEN '2022-01-01' AND '2022-12-31';

--Q4.
SELECT
    SUM(t.Amount) AS TotalDeposits
FROM
    FB.Transactions t
WHERE
    t.TransactionType = 'Deposit'
    AND t.TransactionDate >= '2022-05-01'
    AND t.TransactionDate < '2022-06-01';

--Q5.
SELECT
    l.LoanID,
    l.CustomerID,
    l.LoanType,
    l.LoanAmount,
    l.InterestRate,
    l.LoanDate
FROM
    FB.Loans l
WHERE
    l.CustomerID = 'C0768';

--Q6.
SELECT
    e.EmployeeID,
    e.FirstName,
    e.LastName,
    e.Position,
    e.HireDate,
    e.PhoneNumber,
    e.BranchID
FROM
    FB.Employees e
WHERE
    e.BranchID = 'B0851';


--Q7.
SELECT
     COUNT(cc.CardID) AS TotalCreditCardsIssued
FROM
    FB.Credit_cards cc;

--Q8.
SELECT
     AVG(l.InterestRate) AS AverageInterestRate
FROM
    FB.Loans l;

--Q9.
```

SOLUTIONS

```sql
SELECT DISTINCT
    c.CustomerID,
    c.FirstName,
    c.LastName,
    c.Gender,
    c.DateOfBirth,
    c.Email,
    c.PhoneNumber
FROM
    FB.Customers c
JOIN
    FB.Accounts a
ON
    c.CustomerID = a.CustomerID
JOIN
    FB.Transactions t
ON
    a.AccountID = t.AccountID
WHERE
    t.TransactionDate >= '2020-01-01'
    AND t.TransactionDate < '2021-01-01';

--Q10.
SELECT
    a.AccountID,
    a.CustomerID,
    a.AccountType,
    a.Balance,
    a.OpenDate
FROM
    FB.Accounts a
WHERE
    a.AccountID NOT IN (
        SELECT
            t.AccountID
        FROM
            FB.Transactions t
        WHERE
            t.TransactionDate >= '2019-01-01'
            AND t.TransactionDate < '2024-01-01'
    );

--Q11.
SELECT
    SUM(p.Amount) AS TotalLoanPayments2015
FROM
    FB.Payments p
WHERE
    p.PaymentDate >= '2015-01-01'
    AND p.PaymentDate < '2016-01-01';

--Q12.
SELECT
```

```sql
    i.InvestmentID,
    i.CustomerID,
    c.FirstName,
    c.LastName,
    i.InvestmentType,
    i.Amount,
    i.InvestmentDate
FROM
    FB.Investments i
JOIN
    FB.Customers c
ON
    i.CustomerID = c.CustomerID
WHERE
    i.InvestmentType = 'Mutual Funds';

--13.
SELECT
    a.AccountType,
    COUNT(t.TransactionID) AS TotalTransactions
FROM
    FB.Transactions t
JOIN
    FB.Accounts a
ON
    t.AccountID = a.AccountID
GROUP BY
    a.AccountType;


--Q14.
SELECT
    b.BranchID,
    b.BranchName,
    COUNT(e.EmployeeID) AS NumberOfEmployees
FROM
    FB.Branches b
JOIN
    FB.Employees e
ON
    b.BranchID = e.BranchID
GROUP BY
    b.BranchID,
    b.BranchName;

--Q15.
SELECT
    l.CustomerID,
    c.FirstName,
    c.LastName,
    SUM(l.LoanAmount) AS TotalOutstandingLoanAmount
FROM
    FB.Loans l
```

```
213  JOIN
214      FB.Customers c
215  ON
216      l.CustomerID = c.CustomerID
217  GROUP BY
218      l.CustomerID,
219      c.FirstName,
220      c.LastName;
221
222  --Q16.
223  SELECT DISTINCT
224      c.CustomerID,
225      c.FirstName,
226      c.LastName,
227      c.Gender,
228      c.DateOfBirth,
229      c.Email,
230      c.PhoneNumber
231  FROM
232      FB.Customers c
233  JOIN
234      FB.Accounts a
235  ON
236      c.CustomerID = a.CustomerID
237  WHERE
238      c.CustomerID IN (
239          SELECT
240              CustomerID
241          FROM
242              FB.Accounts
243          GROUP BY
244              CustomerID
245          HAVING
246              COUNT(DISTINCT AccountType) > 1
247      );
248
249
250  --Q17.
251  SELECT
252      COUNT(l.LoanID) AS TotalLoansApproved2017
253  FROM
254      FB.Loans l
255  WHERE
256      l.LoanDate >= '2017-01-01'
257      AND l.LoanDate < '2018-01-01';
258
259
260  --Q18.
261  SELECT
262      AVG(a.Balance) AS AverageSavingsBalance
263  FROM
264      FB.Accounts a
265  WHERE
```

```sql
    a.AccountType = 'Savings';


--Q19.
SELECT DISTINCT
    c.CustomerID,
    c.FirstName,
    c.LastName,
    c.Gender,
    c.DateOfBirth,
    c.Email,
    c.PhoneNumber
FROM
    FB.Customers c
JOIN
    FB.Investments i
ON
    c.CustomerID = i.CustomerID
WHERE
    i.InvestmentType = 'Stocks';


--Q20.
SELECT
    SUM(l.LoanAmount * l.InterestRate / 100) AS TotalInterestEarned2012
FROM
    FB.Loans l
WHERE
    l.LoanDate >= '2012-01-01'
    AND l.LoanDate < '2013-01-01';

--Q21.
SELECT
    COUNT(t.TransactionID) AS TotalDeposits
FROM
    FB.Transactions t
JOIN
    FB.Accounts a
ON
    t.AccountID = a.AccountID
JOIN
    FB.Employees e
ON
    a.CustomerID = e.EmployeeID
JOIN
    FB.Branches b
ON
    e.BranchID = b.BranchID
WHERE
    t.TransactionType = 'Deposit'
    AND b.BranchID = 'B0036';


```

```sql

--Q22.
SELECT
    e.EmployeeID,
    e.FirstName,
    e.LastName,
    e.Position,
    e.HireDate,
    e.PhoneNumber,
    e.BranchID
FROM
    FB.Employees e
WHERE
    e.HireDate >= '2018-01-01'
    AND e.HireDate < '2019-01-01';

--Q23.
SELECT
    SUM(i.Amount) AS TotalInvestments
FROM
    FB.Investments i;

--Q24.
SELECT DISTINCT
    c.CustomerID,
    c.FirstName,
    c.LastName,
    c.Gender,
    c.DateOfBirth,
    c.Email,
    c.PhoneNumber
FROM
    FB.Customers c
JOIN
    (SELECT
         CustomerID
     FROM
         FB.Loans
     GROUP BY
         CustomerID
     HAVING
         COUNT(LoanID) > 1) l
ON
    c.CustomerID = l.CustomerID;

--Q25.
SELECT
    a.AccountID,
    a.CustomerID,
    a.AccountType,
    a.Balance,
    a.OpenDate
FROM
```

```
372      FB.Accounts a
373  WHERE
374      a.Balance < 500;
375
376  --Q26.
377  SELECT
378      SUM(t.Amount) AS TotalWithdrawalsJanuary2023
379  FROM
380      FB.Transactions t
381  WHERE
382      t.TransactionType = 'Withdrawal'
383      AND t.TransactionDate >= '2023-01-01'
384      AND t.TransactionDate < '2023-02-01';
385
386  --Q27.
387  SELECT DISTINCT
388      c.CustomerID,
389      c.FirstName,
390      c.LastName,
391      c.Gender,
392      c.DateOfBirth,
393      c.Email,
394      c.PhoneNumber
395  FROM
396      FB.Customers c
397  JOIN
398      FB.Loans l
399  ON
400      c.CustomerID = l.CustomerID
401  JOIN
402      FB.Payments p
403  ON
404      l.LoanID = p.LoanID
405  WHERE
406      p.PaymentMethod = 'Bank Transfer';
407
408  --Q28.
409  SELECT
410      COUNT(t.TransactionID) AS TotalCreditCardTransactions2022
411  FROM
412      FB.Transactions t
413  JOIN
414      FB.Credit_cards cc
415  ON
416      t.AccountID = cc.CardID
417  WHERE
418      t.TransactionDate >= '2022-01-01'
419      AND t.TransactionDate < '2023-01-01';
420
421  --Q29.
422  SELECT
423      AVG(cc.CreditLimit) AS AverageCreditLimit
424  FROM
```

```
425     FB.Credit_cards cc;
426
427 --Q30.
428 SELECT DISTINCT
429     c.CustomerID,
430     c.FirstName,
431     c.LastName,
432     c.Gender,
433     c.DateOfBirth,
434     c.Email,
435     c.PhoneNumber
436 FROM
437     FB.Customers c
438 JOIN
439     FB.Investments i
440 ON
441     c.CustomerID = i.CustomerID
442 WHERE
443     i.InvestmentType = 'Bonds';
444
445 --Q31.
446 SELECT
447     l.LoanType,
448     COUNT(l.LoanID) AS TotalLoansApproved
449 FROM
450     FB.Loans l
451 GROUP BY
452     l.LoanType;
453
454 --Q32.
455 SELECT
456     e.EmployeeID,
457     e.FirstName,
458     e.LastName,
459     e.Position,
460     e.HireDate,
461     e.PhoneNumber,
462     e.BranchID
463 FROM
464     FB.Employees e
465 WHERE
466     e.Position = 'Loan Officer';
467
468
469 --Q33.
470 SELECT
471     COUNT(a.AccountID) AS TotalAccountsOpened2014
472 FROM
473     FB.Accounts a
474 WHERE
475     a.OpenDate >= '2014-01-01'
476     AND a.OpenDate < '2015-01-01';
477
```

```sql

--Q34.
SELECT
    t.TransactionType,
    AVG(t.Amount) AS AverageTransactionAmount
FROM
    FB.Transactions t
GROUP BY
    t.TransactionType;


--Q35.
WITH CustomerBalances AS (
    SELECT Top 10
        a.CustomerID,
        SUM(a.Balance) AS TotalBalance
    FROM
        FB.Accounts a
    GROUP BY
        a.CustomerID
), CustomerInvestments AS (
    SELECT
        i.CustomerID,
        SUM(i.Amount) AS TotalInvestments
    FROM
        FB.Investments i
    GROUP BY
        i.CustomerID
), CustomerTotalValue AS (
    SELECT
        cb.CustomerID,
        (cb.TotalBalance + COALESCE(ci.TotalInvestments,  0)) AS TotalValue
    FROM
        CustomerBalances cb
    LEFT JOIN
        CustomerInvestments ci
    ON
        cb.CustomerID = ci.CustomerID
)
SELECT
    c.CustomerID,
    c.FirstName,
    c.LastName,
    c.Gender,
    c.DateOfBirth,
    c.Email,
    c.PhoneNumber,
    ct.TotalValue
FROM
    FB.Customers c
JOIN
    CustomerTotalValue ct
ON
```

```
531      c.CustomerID = ct.CustomerID
532  ORDER BY
533      ct.TotalValue DESC
534
535  --Q36.
536  WITH CustomerBalances AS (
537      SELECT
538          a.CustomerID,
539          SUM(a.Balance) AS TotalBalance
540      FROM
541          FB.Accounts a
542      GROUP BY
543          a.CustomerID
544  ), CustomerTransactions AS (
545      SELECT
546          a.CustomerID,
547           COUNT(t.TransactionID) AS TransactionCount
548      FROM
549          FB.Transactions t
550      JOIN
551          FB.Accounts a
552      ON
553          t.AccountID = a.AccountID
554      GROUP BY
555          a.CustomerID
556  ), CustomerInvestments AS (
557      SELECT
558          i.CustomerID,
559          SUM(i.Amount) AS TotalInvestments
560      FROM
561          FB.Investments i
562      GROUP BY
563          i.CustomerID
564  ), CustomerMetrics AS (
565      SELECT
566          cb.CustomerID,
567          cb.TotalBalance,
568           COALESCE(ct.TransactionCount, 0) AS TransactionCount,
569           COALESCE(ci.TotalInvestments, 0) AS TotalInvestments
570      FROM
571          CustomerBalances cb
572      LEFT JOIN
573          CustomerTransactions ct
574      ON
575          cb.CustomerID = ct.CustomerID
576      LEFT JOIN
577          CustomerInvestments ci
578      ON
579          cb.CustomerID = ci.CustomerID
580  )
581  SELECT
582      cm.CustomerID,
583      cm.TotalBalance,
```

```
584      cm.TransactionCount,
585      cm.TotalInvestments,
586      CASE
587          WHEN cm.TotalBalance >= 100000 OR cm.TotalInvestments >= 50000 THEN
               'High Value'
588          WHEN cm.TotalBalance >= 50000 OR cm.TotalInvestments >= 25000 THEN
               'Medium Value'
589          ELSE 'Low Value'
590      END AS CustomerSegment
591  FROM
592      CustomerMetrics cm;
593
594
595  --Q37.
596  -- Calculate Total Deposits and Total Withdrawals for Each Account
597  WITH TotalDeposits AS (
598      SELECT
599          t.AccountID,
600          SUM(t.Amount) AS TotalDepositAmount
601      FROM
602          FB.Transactions t
603      WHERE
604          t.TransactionType = 'Deposit'
605      GROUP BY
606          t.AccountID
607  ), TotalWithdrawals AS (
608      SELECT
609          t.AccountID,
610          SUM(t.Amount) AS TotalWithdrawalAmount
611      FROM
612          FB.Transactions t
613      WHERE
614          t.TransactionType = 'Withdrawal'
615      GROUP BY
616          t.AccountID
617  )
618
619  -- Combine Deposits and Withdrawals with Current Balance for Each Customer
620  SELECT
621      c.CustomerID,
622      c.FirstName,
623      c.LastName,
624      COALESCE(SUM(td.TotalDepositAmount), 0) AS TotalDeposits,
625      COALESCE(SUM(tw.TotalWithdrawalAmount), 0) AS TotalWithdrawals,
626      SUM(a.Balance) AS CurrentBalance
627  FROM
628      FB.Customers c
629  LEFT JOIN
630      FB.Accounts a
631  ON
632      c.CustomerID = a.CustomerID
633  LEFT JOIN
634      TotalDeposits td
```

```sql
ON
    a.AccountID = td.AccountID
LEFT JOIN
    TotalWithdrawals tw
ON
    a.AccountID = tw.AccountID
GROUP BY
    c.CustomerID,
    c.FirstName,
    c.LastName
ORDER BY
    c.CustomerID;


--Q38.
SELECT
    c.CustomerID,
    c.FirstName,
    c.LastName,
    SUM(i.Amount) AS TotalInvestmentAmount,
    STRING_AGG(i.InvestmentType, ', ') AS InvestmentTypes
FROM
    FB.Customers c
JOIN
    FB.Investments i
ON
    c.CustomerID = i.CustomerID
GROUP BY
    c.CustomerID,
    c.FirstName,
    c.LastName
ORDER BY
    TotalInvestmentAmount DESC;


--Q39.
SELECT
    YEAR(t.TransactionDate) AS Year,
    MONTH(t.TransactionDate) AS Month,
    t.TransactionType,
    SUM(t.Amount) AS TotalTransactionVolume
FROM
    FB.Transactions t
WHERE
    t.TransactionDate >= '2011-01-01'
    AND t.TransactionDate < '2024-01-01'
GROUP BY
    YEAR(t.TransactionDate),
    MONTH(t.TransactionDate),
    t.TransactionType
ORDER BY
    Year,
    Month,
```

```sql
    t.TransactionType;


--Q40.
WITH TotalPayments AS (
    SELECT
        p.LoanID,
        SUM(p.Amount) AS TotalRepaid
    FROM
        FB.Payments p
    GROUP BY
        p.LoanID
)
SELECT
    l.LoanID,
    l.CustomerID,
    l.LoanAmount,
    COALESCE(tp.TotalRepaid, 0) AS TotalRepaid,
    (l.LoanAmount - COALESCE(tp.TotalRepaid, 0)) AS OutstandingBalance,
    CASE
        WHEN COALESCE(tp.TotalRepaid, 0) >= l.LoanAmount THEN 'Fully Paid'
        WHEN COALESCE(tp.TotalRepaid, 0) = 0 THEN 'Not Started'
        ELSE 'In Progress'
    END AS RepaymentStatus
FROM
    FB.Loans l
LEFT JOIN
    TotalPayments tp
ON
    l.LoanID = tp.LoanID
ORDER BY
    l.LoanID;


--Q41.
WITH YearlyInvestments AS (
    SELECT
        YEAR(i.InvestmentDate) AS Year,
        SUM(i.Amount) AS TotalInvestmentAmount
    FROM
        FB.Investments i
    GROUP BY
        YEAR(i.InvestmentDate)
)
SELECT
    y1.Year,
    y1.TotalInvestmentAmount,
    (y1.TotalInvestmentAmount - y2.TotalInvestmentAmount) /
      y2.TotalInvestmentAmount * 100 AS YoYGrowth
FROM
    YearlyInvestments y1
LEFT JOIN
    YearlyInvestments y2
```

```sql
ON
    y1.Year = y2.Year + 1
ORDER BY
    y1.Year;


--Q42.
SELECT
    c.CustomerID,
    c.FirstName,
    c.LastName,
    c.Email,
    cc.CardID,
    cc.CardNumber,
    cc.ExpirationDate
FROM
    FB.Customers c
JOIN
    FB.Credit_cards cc
ON
    c.CustomerID = cc.CustomerID
WHERE
    cc.ExpirationDate BETWEEN '2022-01-01' AND DATEADD(MONTH, 3,
      '2022-01-01')
ORDER BY
    cc.ExpirationDate;


--Q43.
WITH CustomerInvestments AS (
    SELECT
        i.CustomerID,
        i.InvestmentType,
        SUM(i.Amount) AS TotalInvestmentAmount
    FROM
        FB.Investments i
    GROUP BY
        i.CustomerID,
        i.InvestmentType
)
SELECT
    c.CustomerID,
    c.FirstName,
    c.LastName,
    ci.InvestmentType,
    ci.TotalInvestmentAmount,
    a.Balance AS AccountBalance
FROM
    FB.Customers c
JOIN
    CustomerInvestments ci
ON
    c.CustomerID = ci.CustomerID
```

```
792  JOIN
793      FB.Accounts a
794  ON
795      c.CustomerID = a.CustomerID
796  ORDER BY
797      c.CustomerID,
798      ci.InvestmentType;
799
800  --Q44.
801  WITH BranchAccounts AS (
802      SELECT
803          a.BranchID,
804          COUNT(a.AccountID) AS TotalAccounts,
805          SUM(a.Balance) AS TotalBalance
806      FROM
807          FB.Accounts a
808      GROUP BY
809          a.BranchID
810  ), BranchLoans AS (
811      SELECT
812          l.BranchID,
813          SUM(l.LoanAmount) AS TotalLoanAmount
814      FROM
815          FB.Loans l
816      GROUP BY
817          l.BranchID
818  ), BranchTransactions AS (
819      SELECT
820          a.BranchID,
821          SUM(t.Amount) AS TotalTransactionAmount
822      FROM
823          FB.Transactions t
824      JOIN
825          FB.Accounts a
826      ON
827          t.AccountID = a.AccountID
828      GROUP BY
829          a.BranchID
830  )
831  SELECT
832      b.BranchID,
833      b.BranchName,
834      COALESCE(ba.TotalAccounts, 0) AS TotalAccounts,
835      COALESCE(ba.TotalBalance, 0) AS TotalBalance,
836      COALESCE(bl.TotalLoanAmount, 0) AS TotalLoanAmount,
837      COALESCE(bt.TotalTransactionAmount, 0) AS TotalTransactionAmount,
838      COUNT(e.EmpID) AS TotalEmployees
839  FROM
840      FB.Branches b
841  LEFT JOIN
842      BranchAccounts ba
843  ON
844      b.BranchID = ba.BranchID
```

```
845  LEFT JOIN
846      BranchLoans bl
847  ON
848       b.BranchID = bl.BranchID
849  LEFT JOIN
850      BranchTransactions bt
851  ON
852       b.BranchID = bt.BranchID
853  LEFT JOIN
854      FB.Employees e
855  ON
856      b.BranchID = e.BranchID
857  GROUP BY
858      b.BranchID,
859      b.BranchName,
860      ba.TotalAccounts,
861      ba.TotalBalance,
862      bl.TotalLoanAmount,
863       bt.TotalTransactionAmount
864  ORDER BY
865      b.BranchName;
866
```

Appendix

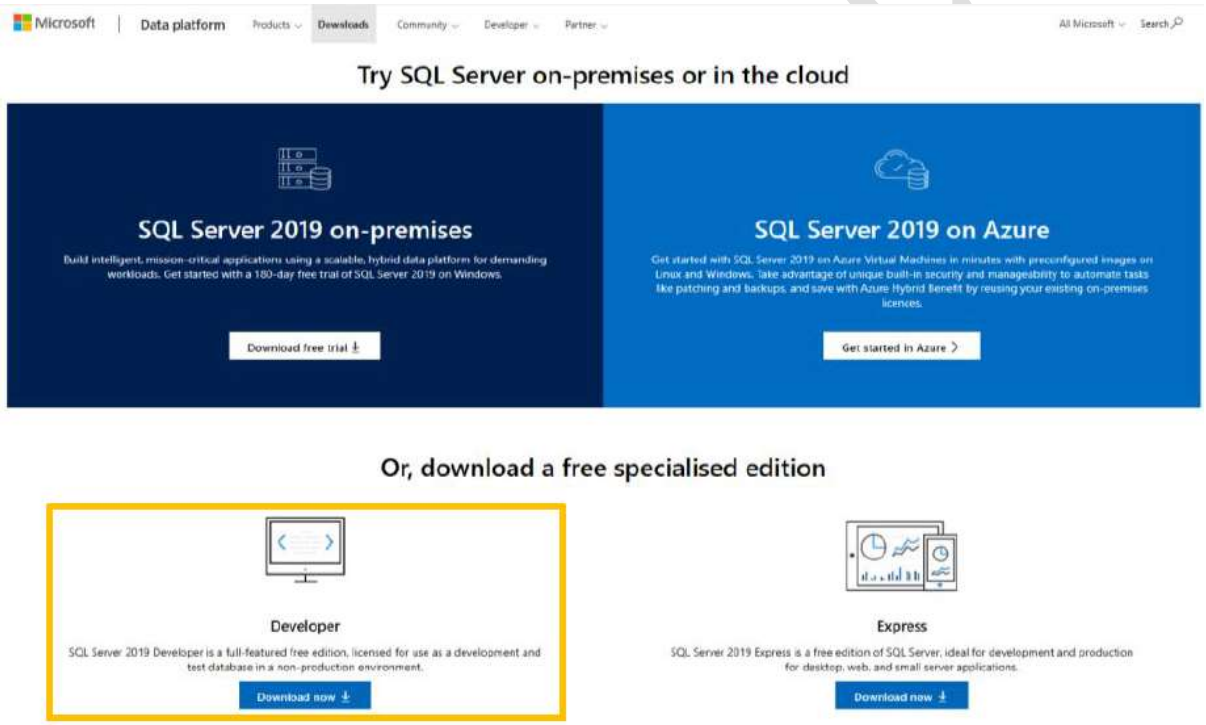
SQL Server Installation Guide

Phase 1: Installing SQL Server 2019 on a Windows 10 operating system.

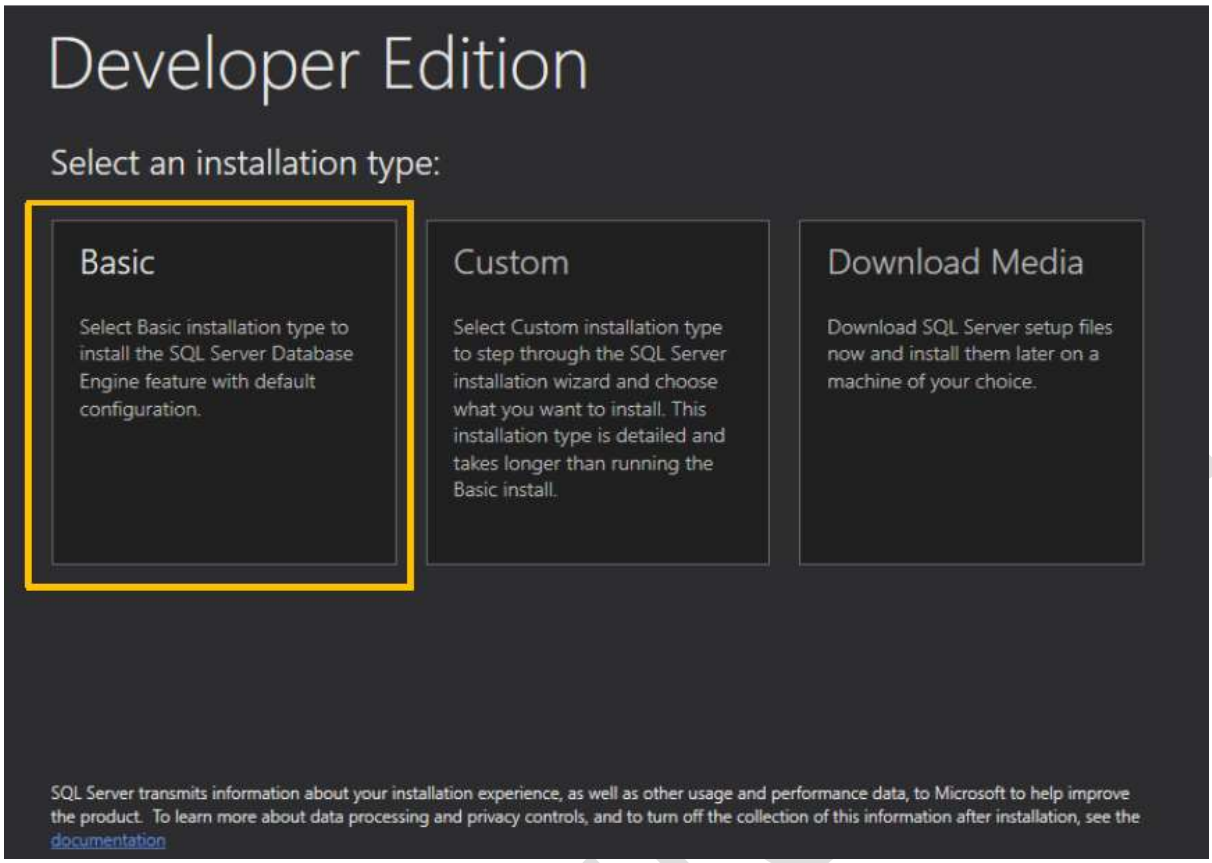
1. To set up SQL Server 2019, obtain the necessary files by clicking the provided link:

   https://www.microsoft.com/en-gb/sql-server/sql-server-downloads

2. Select "Download Now" for the developer edition.



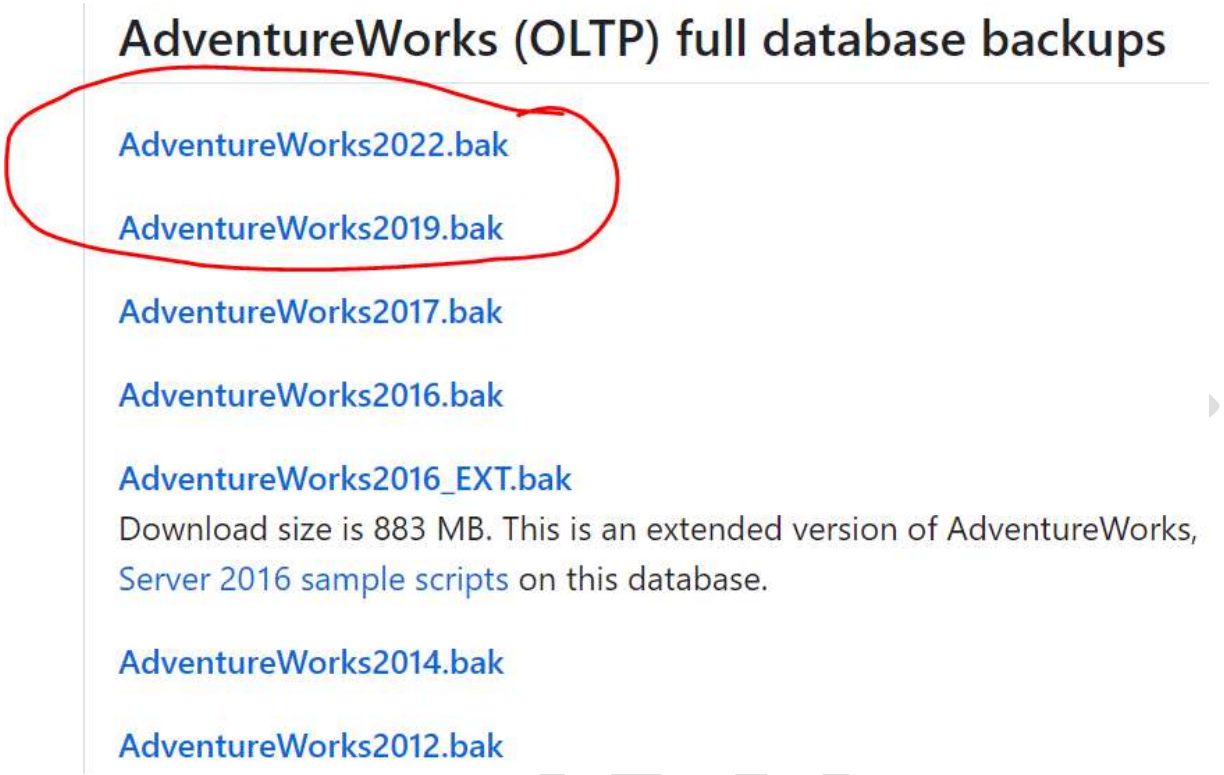
3. After the file has finished downloading, double-click on it to initiate the installation.
4. In the window that appears, choose the "Basic" installation type.



5. Press "Next," agree to the Terms and Conditions, and then click "Install."
6. After the installation is finished, you will receive a link to download SQL Server Management Studio. If you don't spot the link, please click on this provided link: https://aka.ms/ssmsfullsetup
7. Download SQL Server Management Studio and proceed to install it.

Phase 2: AdventureWorks (2019 or 2022) Database

1. Upon the successful installation of SQL Server 2019, you'll require a database for practice. Please follow the link below to download the AdventureWorks2019 or 2022 Database: https://github.com/Microsoft/sql-server-samples/releases/tag/adventureworks
2. On the webpage, locate and select the highlighted option to download the AdventureWorks2019.bak file or AdventureWorks2022.bak file.



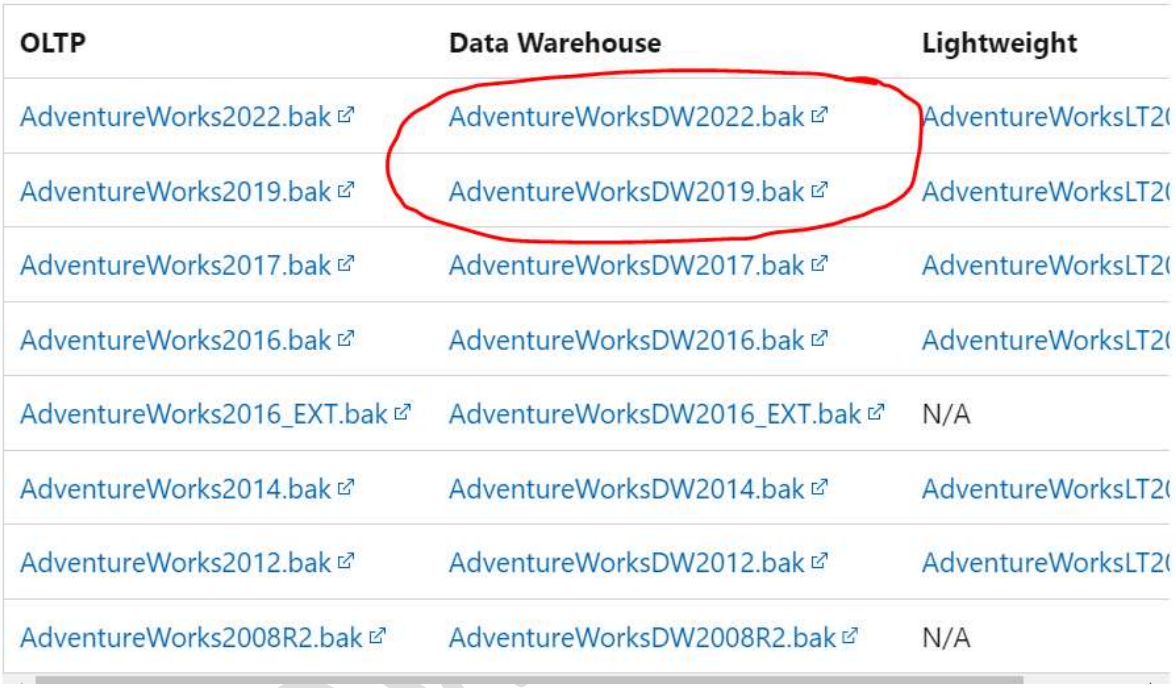
3. Navigate to the folder where the AdventureWorks2012.bak file has been downloaded and proceed to make a copy of the file.
4. Paste the file into the Backup folder within your freshly installed SQL system, which should be situated in a location resembling the one described below:

   C:\Program Files\Microsoft SQL Server\MSSQL11.SQLSERVERBI\MSSQL\Backup
5. Next, open SQL Server Management Studio from either the Programs Menu or the Applications Desktop (Windows 8).
6. Now click on the following link to restore the database on SQL Server:

   https://learn.microsoft.com/en-us/sql/samples/adventureworks-install-configure?view=sql-server-ver16&tabs=ssms

Phase 3: AdventureWorks Data Warehouse Version (2019 or 2022) Database

1. The following link allows you to download the AdventureWorksDW2019.bak or AdventureWorksDW2022.bak versions: https://learn.microsoft.com/en-us/sql/samples/adventureworks-install-configure?view=sql-server-ver16&tabs=ssms
2. After downloading the file, following similar steps as in **Phase 2** to restore the database.



OLTP	Data Warehouse	Lightweight
AdventureWorks2022.bak	AdventureWorksDW2022.bak	AdventureWorksLT2(
AdventureWorks2019.bak	AdventureWorksDW2019.bak	AdventureWorksLT2(
AdventureWorks2017.bak	AdventureWorksDW2017.bak	AdventureWorksLT2(
AdventureWorks2016.bak	AdventureWorksDW2016.bak	AdventureWorksLT2(
AdventureWorks2016_EXT.bak	AdventureWorksDW2016_EXT.bak	N/A
AdventureWorks2014.bak	AdventureWorksDW2014.bak	AdventureWorksLT2(
AdventureWorks2012.bak	AdventureWorksDW2012.bak	AdventureWorksLT2(
AdventureWorks2008R2.bak	AdventureWorksDW2008R2.bak	N/A

Phase 4: WideWorldImporters Database:

1. The link to download the WideWorldImporters database is: https://github.com/Microsoft/sql-server-samples/releases/tag/wide-world-importers-v1.0
2. Ensure to following the steps as outlined in Phase 2 to restore the database.

▾ Assets 17

Daily.ETL.ispac	61.2 KB	Aug 12, 2016
sample-scripts.zip	23.1 KB	Jun 8, 2016
WideWorldImporters-Full.bacpac	58.5 MB	Oct 7, 2022
WideWorldImporters-Full.bak	121 MB	Oct 7, 2022
WideWorldImporters-Full_old.bacpac	59.1 MB	Nov 16, 2016
WideWorldImporters-Full_old.bak	121 MB	Aug 13, 2016
WideWorldImporters-Standard.bacpac	58.2 MB	Oct 7, 2022
WideWorldImporters-Standard.bak	121 MB	Oct 7, 2022
WideWorldImporters-Standard_old.bacpac	58.5 MB	Jun 8, 2016
WideWorldImporters-Standard_old.bak	121 MB	Aug 15, 2016
WideWorldImportersDW-Full.bacpac	19.6 MB	Nov 16, 2016
WideWorldImportersDW-Full.bak	47.7 MB	Jun 8, 2016
WideWorldImportersDW-Standard.bacpac	21.4 MB	Jun 8, 2016
WideWorldImportersDW-Standard.bak	51.4 MB	Jun 8, 2016