

# Database Technologies – UE20CS343

## Assignment – 2

Name: Vishwa Mehul Mehta

SRN: PES2UG20CS389

Sec: F

Date: 19-03-2023

### Random Value Insertions:

The screenshot shows the Microsoft SQL Server Management Studio interface. The left pane displays the Object Explorer with the connection to 'ACER-VISHWA\SQLEXPRESS01'. The right pane shows a query window titled 'SQLQuery1.sql - A...VISHWA\Vishwa (73)\*' containing the following T-SQL script:

```
set nocount on
Declare @count int = 13
while(@count<=2010)
Begin
    Declare @cust_id int = @count
    Declare @phone_no int = @count
    Declare @cust_email_id nvarchar(50)='ABC'+ltrim(@count)+'@gmail.com'
    Declare @cust_name varchar(50) = 'ABC'
    Declare @str_no int = [data type varchar(50)] set
    Declare @str_name varc[50] set
    Declare @pincode int = 567938
    insert into customer values(@cust_id, @phone_no, @cust_email_id, @cust_name, @str_no, @str_name, @pincode)
    set @count=@count+1
End
```

The results pane shows a table with 255 rows of inserted data. The columns are: cust\_id, phone\_no, cust\_email\_id, cust\_name, str\_no, str\_name, pincode. The data includes various names like Vinay Aravind, Anil Abhishek, Indepal Ravinder, Sunaina Mishra, etc., and addresses like Chakraborty Street, Shroff Street, Goel Street, etc.

cust_id	phone_no	cust_email_id	cust_name	str_no	str_name	pincode
6	43143	5636459498	vinayaravind24@hotmail.com	26	Chakraborty Street	525343
7	45406	4305436406	anilabhik@gmail.com	71	Shroff Street	234740
8	57317	5289998172	inderpal@gmail.com	12	Goel Street	649721
9	57652	1234567890	sunaina@gmail.com	12	Shroff Street	234740
10	58462	8545588678	anik@gmail.com	56	Sekhon Street	139845
11	63698	3816889497	ninadram@gmail.com	37	Kari Street	456639
12	76443	7294778333	pravinjaya@gmail.com	22	Raja Street	591648
13	97	97	ABC97@gmail.com	ABC	ABC street	567938
14	98	98	ABC98@gmail.com	98	ABC street	567938
15	99	99	ABC99@gmail.com	ABC	ABC street	567938
16	100	100	ABC100@gmail.com	100	ABC street	567938
17	101	101	ARC101@gmail.com	101	ARC street	567938

SQLQuery1.sql - ACER-VISHWA\SQLEXPRESS01.inventory\_management\_system (ACER-VISHWA\Vishwa (73)) - Microsoft SQL Server Management Studio

```

set nocount on
Declare @count int = 9
while(@count<=2010)
Begin
    Declare @pid nvarchar(50) = @count
    Declare @purchase_date varchar(50) = '2023-03-18'
    Declare @mfg_date varchar(50) = '2023-03-14'
    Declare @pqty int = @count
    insert into stock values(@pid, @purchase_date, @mfg_date, @pqty)
    set @count=@count+1
End
select * from stock;

```

Results Messages

pid	purchase_date	mfg_date	pqty
1	2023-03-18	2023-03-14	9
2	2023-03-18	2023-03-14	10
3	2023-03-18	2023-03-14	11
4	2023-03-18	2023-03-14	12
5	2023-03-18	2023-03-14	13
6	2023-03-18	2023-03-14	14
7	2023-03-18	2023-03-14	15
8	2023-03-18	2023-03-14	16
9	2023-03-18	2023-03-14	17
10	2023-03-18	2023-03-14	18
11	2023-03-18	2023-03-14	19
12	2023-03-18	2023-03-14	20
13	2023-03-19	2023-03-14	21

Query completed with errors.

SQLQuery1.sql - ACER-VISHWA\SQLEXPRESS01.inventory\_management\_system (ACER-VISHWA\Vishwa (73)) - Microsoft SQL Server Management Studio

```

set nocount on
Declare @count int = 12
while(@count<=2018)
Begin
    Declare @fname varchar(50) = 'abcd'
    Declare @lname varchar(50) = 'adfg'
    Declare @vendor_id int = @count
    Declare @email_id varchar(50) = 'ABC'+ltrim(@count)+'@gmail.com'
    Declare @prod_id int = @count
    Declare @cust_id int = @count
    insert into vendor values(@fname, @lname, @vendor_id, @email_id, @prod_id, @cust_id)
    set @count=@count+1
End
select * from vendor;

```

Results Messages

fname	lname	vendor_id	email_id	product_id	cust_id
ranga...	s	94138	rangeela@gmail.c...	45124	26864
ranga...	s	94138	rangeela@gmail.c...	98390	45406
rakesh	sharma	29690	rakesh@gmail.com	81010	43143
rakesh	sharma	29690	rakesh@gmail.com	15292	43143
vishwa	mehta	73660	vis@gmail.com	59854	58462
visma...	r	84992	vismaya@gmail.c...	NULL	NULL
neeta	sharma	40226	neeta@gmail.com	NULL	NULL
Mehul	Mehta	74044	msm124@yahoo...	59854	13261
vishwa	mehta	73660	vis@gmail.com	53028	57652

Query executed successfully.

SQLQuery1.sql - ACER-VISHWA\SQLEXPRESS01.inventory\_management\_system (ACER-VISHWA\Vishwa (73)) - Microsoft SQL Server Management Studio

```

set nocount on
Declare @count int = 9
while(@count<=2010)
Begin
    Declare @prod_id int = @count
    Declare @invoice_no int = @count
    Declare @invoice_date varchar(50) = '2023-03-18'
    Declare @selling_price float = @count * 0.8
    Declare @prod_qty int = @count
    Declare @discount float = 0.2
    Declare @phone_no int = 1234567890
    Declare @vendor_id int = @count
    insert into invoice values(@prod_id, @invoice_no, @invoice_date, @selling_price, @prod_qty, @discount, @phone_no, @vendor_id)
    set @count=@count+1
End
select * from invoice;

```

Results Messages

prod_id	invoice_no	invoice_date	selling_price	prod_qty	discount	phone_no	vendor_id
1	9	2023-03-18	7.2	9	0.2	1234567890	9
2	10	2023-03-18	8	10	0.2	1234567890	10
3	11	2023-03-18	8.8	11	0.2	1234567890	11
4	12	2023-03-18	9.6	12	0.2	1234567890	12
5	13	2023-03-18	10.4	13	0.2	1234567890	13
6	14	2023-03-18	11.2	14	0.2	1234567890	14
7	15	2023-03-18	12	15	0.2	1234567890	15
8	16	2023-03-18	12.8	16	0.2	1234567890	16
9	17	2023-03-18	13.6	17	0.2	1234567890	17

Query completed with errors.

## (a) Task 1:

1.

SQLQuery1.sql - ACER-VISHWA\SQLEXPRESS01.inventory\_management\_system (ACER-VISHWA\Vishwa (73)) - Microsoft SQL Server Management Studio

```

select * from invoice INNER JOIN product on invoice.prod_id = product.product_id
INNER JOIN vendor on invoice.prod_id = vendor.product_id;

```

Messages Execution plan

Query 1: Query cost (relative to the batch): 100%

select \* from invoice INNER JOIN product on invoice.prod\_id = product.product\_id INNER JOIN vendor on invoice.pro...

```

graph TD
    SELECT[SELECT] -->|Cost: 0 %| NESTED[Inner Join]
    NESTED -->|Cost: 2 %| HASH[Hash Match]
    HASH -->|Cost: 46 %| CI_S[Clustered Index Scan]
    CI_S -->|Cost: 5 %| PK_PRODUCT[product].[PK_product]
    PK_PRODUCT -->|Cost: 18 %| CI_INVOICE[Clustered Index Seek]
    CI_INVOICE -->|Cost: 29 %| TABLE_SCAN[Table Scan]
    TABLE_SCAN -->|Cost: 29 %| PK_VENDOR[Vendor].[PK_vendor]

```

Query executed successfully.

The screenshot shows the Microsoft SQL Server Management Studio interface. The Object Explorer on the left lists the database structure, including tables like invoice, product, and vendor. The SQL Query window contains a query:

```
select * from invoice INNER JOIN product on invoice.prod_id = product.product_id  
INNER JOIN vendor on invoice.vendor_id = vendor.vendor_id;
```

An execution plan is displayed in the center, showing a "Nested Loops" operation. The plan details the following steps:

- Physical Operation**: Nested Loops
- Logical Operation**: Inner Join
- Estimated Execution Mode**: Row
- Estimated I/O Cost**: 0
- Estimated Operator Cost**: 0.0010701 (2%)
- Estimated CPU Cost**: 0.0010701
- Estimated Subtree Cost**: 0.06971256
- Estimated Number of Executions**: 1
- Estimated Number of Rows Per Execution**: 256
- Estimated Number of Rows for All Executions**: 256
- Estimated Row Size**: 331 B
- Node ID**: 0

The execution plan also shows the following details for the inner join:

- product.product\_id INNER JOIN vendor on invoice.pro...**
- Index Scan (C...)**
  - 1. [PK\_product]**  
Cardinality: 5 %
- Table Scan [vendor]**  
Cardinality: 29 %

The status bar at the bottom indicates "Query executed successfully".

2.

The screenshot shows the Microsoft SQL Server Management Studio interface. The title bar reads "SQLQuery1.sql - ACER-VISHWA\SQLEXPRESS01.inventory\_management\_system (ACER-VISHWA\Vishwa (73)) - Microsoft SQL Server Management Studio". The Object Explorer sidebar on the left lists the database structure, including the "inventory\_management\_system" database with its tables: customer, invoice, login, product, stock, and vendor. The main pane displays a query window with the following T-SQL code:

```
select * from invoice INNER JOIN stock on invoice.prod_id = stock.pid  
INNER JOIN vendor on invoice.prod_id = vendor.product_id;
```

Below the code, the "Execution plan" tab is selected, showing a detailed diagram of the query's execution flow. The plan starts with a "SELECT" node (Cost: 0 %) which performs a "Hash Match (Inner Join)" (Cost: 46 %) with a "Table Scan [vendor] (Cost: 21 %)". This joins with another "Hash Match (Inner Join)" (Cost: 24 %) with a "Clustered Index Scan C\_ [stock].[PK\_stock]" (Cost: 4 %). Finally, it joins with a third "Clustered Index Scan C\_ [invoice].[PK\_invoice]" (Cost: 5 %).

The screenshot shows the Microsoft SQL Server Management Studio interface. The Object Explorer on the left lists the database structure, including the 'inventory\_management\_system' database. The central pane displays a query window titled 'SQLQuery1.sql - A. ISHWA\...'. The query is:

```
select * from [inventory].[dbo].[invoice] inner join vendor on invoice.vendor_id = vendor.vendor_id
```

The execution plan details the following information:

Physical Operation	Hash Match
Logical Operation	Inner Join
Estimated Execution Mode	Row
Estimated Operator Cost	0.0430908 (46%)
Estimated I/O Cost	0
Estimated Subtree Cost	0.0945891
Estimated CPU Cost	0.0430877
Estimated Number of Executions	1
Estimated Number of Rows Per Execution	256
Estimated Number of Rows for All Executions	256
Estimated Row Size	230 B
Node ID	0

The 'Output List' section shows the columns selected from the tables:

- [inventory\_management\_system].[dbo].[invoice].prod\_id,
- [inventory\_management\_system].[dbo].[invoice].invoice\_no,
- [inventory\_management\_system].[dbo].[invoice].invoice\_date,
- [inventory\_management\_system].[dbo].[invoice].selling\_price,
- [inventory\_management\_system].[dbo].[invoice].prod\_qty,
- [inventory\_management\_system].[dbo].[invoice].discount,
- [inventory\_management\_system].[dbo].[invoice].phone\_no,
- [inventory\_management\_system].[dbo].[invoice].vendor\_id,
- [inventory\_management\_system].[dbo].s...

The 'Hash Keys Probe' section lists the primary key columns:

- [inventory\_management\_system].[dbo].[vendor].product\_id
- [inventory\_management\_system].[dbo].[vendor].[product\_id]= [inventory\_management\_system].[dbo].[stock].[pid]

The status bar at the bottom indicates 'Query executed successfully'.

(b) Task 2:

#### 1. LEFT OUTER JOIN – invoice, stock, vendor

The screenshot shows the Microsoft SQL Server Management Studio interface. The title bar indicates the connection is to ACER-VISHWA\SQLEXPRESS01's inventory\_management\_system database. The Object Explorer sidebar shows the database structure, including tables like customer, invoice, login, product, stock, and vendor. The main window displays a query in the SQL Query Editor:

```
select * from invoice LEFT OUTER JOIN stock on invoice.prod_id = stock.pid
LEFT OUTER JOIN vendor on invoice.prod_id = vendor.product_id;
```

Below the query, the 'Execution plan' section shows the cost-based optimizer's plan for the query. The plan consists of three main operations:

- SELECT**: Cost: 0 %
- Hash Match (Left Outer Join)**: Cost: 47 %
- Hash Match (Right Outer Join)**: Cost: 23 %

These two join operations are connected to a **Table Scan [vendor]** (Cost: 21 %) via their respective right-hand sides. Both join operations also connect to a **Clustered Index Scan [stock].[PK\_stock]** (Cost: 4 %) via their left-hand sides. Finally, the **Clustered Index Scan [stock].[PK\_stock]** connects to a **Clustered Index Scan [invoice].[PK\_invoice]** (Cost: 5 %).

At the bottom of the screen, a message states "Query executed successfully." and the status bar shows the session details: ACER-VISHWA\SQLEXPRESS01 (1...) | ACER-VISHWA\Vishwa (73) | inventory\_management\_s... | 00:00:00 | 0 rows.

The screenshot shows the Microsoft SQL Server Management Studio interface. The Object Explorer on the left lists the database structure, including the 'inventory\_management\_system' database. The central pane displays a query window with the following content:

```
SQLQuery1.sql - A_ISHWA\Vi... - [View]
select * from invoice
    LEFT OUTER JOIN vendor
    ON invoice.vendor_id = vendor.vendor_id
```

The 'Physical Operation' section of the execution plan details the 'Hash Match' operation, which uses a 'Left Outer Join'. The estimated cost for the query is 0.0450856 (47%). The output list includes columns from both the 'invoice' and 'vendor' tables.

Physical Operation: Hash Match  
Logical Operation: Left Outer Join  
Estimated Execution Mode: Row  
Estimated Operator Cost: 0.0450856 (47%)  
Estimated I/O Cost: 0  
Estimated Subtree Cost: 0.0965849  
Estimated CPU Cost: 0.0450826  
Estimated Number of Executions: 1  
Estimated Number of Rows Per Execution: 2026  
Estimated Number of Rows for All Executions: 2026  
Estimated Row Size: 230 B  
Node ID: 0

Output List:  
[inventory\_management\_system].[dbo].[invoice].prod\_id,  
[inventory\_management\_system].[dbo].[invoice].invoice\_no,  
[inventory\_management\_system].[dbo].[invoice].invoice\_date,  
[inventory\_management\_system].[dbo].[invoice].selling\_price,  
[inventory\_management\_system].[dbo].[invoice].prod\_qty,  
[inventory\_management\_system].[dbo].[invoice].discount,  
[inventory\_management\_system].[dbo].[invoice].phone\_no,  
[inventory\_management\_system].[dbo].[invoice].vendor\_id,  
[inventory\_management\_system].[dbo].[s...]

Hash Keys Probe  
[inventory\_management\_system].[dbo].[vendor].product\_id  
Probe Residual  
[inventory\_management\_system].[dbo].[invoice].[prod\_id]=  
[inventory\_management\_system].[dbo].[vendor].[product\_id]

## 2. LEFT OUTER JOIN – vendor, invoice, stock

The screenshot shows the Microsoft SQL Server Management Studio interface. The Object Explorer on the left lists the database structure, including the 'inventory\_management\_system' database with its tables: vendor, invoice, and stock. The 'Messages' tab in the center displays the query:

```
select * from vendor LEFT OUTER JOIN invoice on invoice.prod_id = vendor.product_id  
LEFT OUTER JOIN stock on invoice.prod_id = stock.pid;
```

Below the query, the 'Execution plan' window shows the cost-based optimizer's plan. The plan starts with a 'SELECT' node (Cost: 0 %) performing a 'Hash Match' (Right Outer Join) with a 'Clustered Index Scan' on the 'stock' table (Cost: 3 %). This is followed by another 'Hash Match' (Right Outer Join) with a 'Clustered Index Scan' on the 'invoice' table (Cost: 4 %). Finally, it performs a 'Table Scan' on the 'vendor' table (Cost: 16 %).

SQLQuery1.sql - ACER-VISHWA\SQLEXPRESS01.inventory\_management\_system (ACER-VISHWA\Vishwa (73)) - Microsoft SQL Server Management Studio

```
select * from vendor
LEFT OUTER JOIN stock
on vendor.product_id = stock.pid
```

**Hash Match**  
Use each row from the top input to build a hash table, and each row from the bottom input to probe into the hash table, outputting all matching rows.

**Physical Operation** Hash Match  
**Logical Operation** Right Outer Join  
**Estimated Execution Mode** Row  
**Estimated Operator Cost** 0.0567176 (43%)  
**Estimated I/O Cost** 0  
**Estimated Subtree Cost** 0.130995  
**Estimated CPU Cost** 0.0567149  
**Estimated Number of Executions** 1  
**Estimated Number of Rows Per Execution** 7574.87  
**Estimated Number of Rows for All Executions** 7574.87  
**Estimated Row Size** 230 B  
**Node ID** 0

**Output List**  
[inventory\_management\_system].[dbo].[vendor].fname,  
[inventory\_management\_system].[dbo].[vendor].lname,  
[inventory\_management\_system].[dbo].[vendor].vendor\_id,  
[inventory\_management\_system].[dbo].[vendor].email\_id,  
[inventory\_management\_system].[dbo].[vendor].product\_id,  
[inventory\_management\_system].[dbo].[vendor].cust\_id,  
[inventory\_management\_system].[dbo].[invoice].prod\_id,  
[inventory\_management\_system].[dbo].[invoice].invoice\_no,  
[inventory\_management\_system].[dbo].[invoice].invoice\_date...

**Hash Keys Probe**  
[inventory\_management\_system].[dbo].[invoice].prod\_id

**Probe Residual**  
[inventory\_management\_system].[dbo].[invoice].prod\_id = [inventory\_management\_system].[dbo].[stock].pid

**Messages** 0 Execution plan

**Query 1: Query cost**

**SELECT** Cost: 0 %

**Hash Match (Left Outer Join)** Cost: 31 %

**Clustered Index Scan (C... [stock].[PK\_stock]** Cost: 3 %

**Hash Match (Left Outer Join)** Cost: 41 %

**Clustered Index Scan (C... [invoice].[PK\_invoice]** Cost: 5 %

**Table Scan [vendor]** Cost: 16 %

Query executed successfully.

ACER-VISHWA\Vishwa (73) inventory\_management\_s... 00:00:00 0 rows

### 3. RIGHT OUTER JOIN – vendor, invoice, stock

SQLQuery1.sql - ACER-VISHWA\SQLEXPRESS01.inventory\_management\_system (ACER-VISHWA\Vishwa (73)) - Microsoft SQL Server Management Studio

```
select * from vendor RIGHT OUTER JOIN invoice on invoice.prod_id = vendor.product_id
RIGHT OUTER JOIN stock on invoice.prod_id = stock.pid;
```

**Messages** 0 Execution plan

**Query 1: Query cost (relative to the batch): 100%**

**select \* from vendor RIGHT OUTER JOIN invoice on invoice.prod\_id = vendor.product\_id RIGHT OUTER JOIN stock on in...**

**SELECT** Cost: 0 %

**Hash Match (Left Outer Join)** Cost: 31 %

**Clustered Index Scan (C... [stock].[PK\_stock]** Cost: 3 %

**Hash Match (Left Outer Join)** Cost: 41 %

**Clustered Index Scan (C... [invoice].[PK\_invoice]** Cost: 5 %

**Table Scan [vendor]** Cost: 20 %

Query executed successfully.

ACER-VISHWA\SQLEXPRESS01 (1...) ACER-VISHWA\Vishwa (73) inventory\_management\_s... 00:00:00 0 rows

The screenshot shows the Microsoft SQL Server Management Studio interface. The Object Explorer on the left lists the database structure for 'ACER-VISHWA\SQLEXPRESS01'. The main pane displays the results of a query:

```
select * from vendor RIGHT OUTER JOIN invoice on invoice.prod_id = vendor.product_id  
RIGHT OUTER JOIN stock on invoice.prod_id = stock.pid;
```

An execution plan window is open, showing the following details:

Physical Operation	Logical Operation	Cost
Hash Match	Left Outer Join	0.0320223 (31%)
Estimated I/O Cost	Row	0
Estimated Subtree Cost	Row	0.10393
Estimated CPU Cost	Row	0.0320194
Estimated Number of Executions	1	
Estimated Number of Rows per Execution	2026	
Estimated Number of Rows for All Executions	2026	
Estimated Row Size	230 B	
Node ID	0	

The execution plan also includes a 'Table Scan' for the 'vendor' table with a cost of 20%.

#### 4. RIGHT OUTER JOIN – invoice, stock, vendor

The screenshot shows the Microsoft SQL Server Management Studio interface. The Object Explorer on the left lists the database structure, including the inventory\_management\_system database with its tables: customer, invoice, login, product, stock, and vendor. The SQL Query window on the right displays the following T-SQL code:

```
select * from stock RIGHT OUTER JOIN invoice on invoice.prod_id = stock.pid
RIGHT OUTER JOIN vendor on invoice.prod_id = vendor.product_id;
```

Below the code, the execution plan is shown as a tree diagram. The root node is a SELECT statement with a cost of 0 %. It branches into two Hash Match (Right Outer Join) operations. Each of these joins has a cost of 48 %. These two operations then join together via another Hash Match (Right Outer Join) operation with a cost of 23 %. Finally, this result is joined with a Clustered Index Scan on the [stock].[PK\_stock] index with a cost of 4 %. Additionally, there is a Table Scan on the [vendor] table with a cost of 21 %, which is also joined with the main result via a Clustered Index Scan on the [invoice].[PK\_invoice] index with a cost of 5 %.

SQLQuery1.sql - ACER-VISHWA\SQLEXPRESS01.inventory\_management\_system (ACER-VISHWA\Vishwa (73)) - Microsoft SQL Server Management Studio

```
File Edit View Query Project Tools Window Help
File Object Explorer Tools Database Diagrams
[+] ACER-VISHWA\SQLEXPRESS01 (SQL Server 11)
    Databases
        System Databases
        Database Snapshots
        inventory_management_system
        Database Diagrams
    Tables
        System Tables
        FileTables
        External Tables
        Graph Tables
        dbo.customer
        dbo.invoice
        dbo.login
        dbo.product
        dbo.stock
        dbo.vendor
    Views
    External Resources
    Synonyms
    Programmability
    Query Store
    Service Broker
    Storage
    Security
    Ready
```

Object Explorer

SQLQuery1.sql - A...JISHWA\Vis... Hash Match

```
select * from stock
RIGHT OUTER JOIN vendor
ON stock.pid = vendor.product_id;
```

Physical Operation Hash Match

Logical Operation Right Outer Join

Estimated Execution Mode Row

Estimated Operator Cost 0.0474518 (48%)

Estimated I/O Cost 0

Estimated Subtree Cost 0.0989511

Estimated CPU Cost 0.0474488

Estimated Number of Executions 1

Estimated Number of Rows Per Execution 3795

Estimated Number of Rows for All Executions 3795

Estimated Row Size 230 B

Node ID 0

Output List

```
[inventory_management_system].[dbo].[stock].pid,
[inventory_management_system].[dbo].[stock].purchase_date,
[inventory_management_system].[dbo].[stock].mfg_date,
[inventory_management_system].[dbo].[stock].pqty,
[inventory_management_system].[dbo].[invoice].prod_id,
[inventory_management_system].[dbo].[invoice].invoice_no,
[inventory_management_system].[dbo].[invoice].invoice_date,
[inventory_management_system].[dbo].[invoice].selling_price,
[inventory_management_system].[dbo].[invoice].prod_id,
```

Hash Keys Probe

```
[inventory_management_system].[dbo].[vendor].product_id
```

Probe Residual

```
[inventory_management_system].[dbo].[invoice].prod_id =
[inventory_management_system].[dbo].[vendor].product_id]
```

Messages 1# Execution plan

Query 1: Query cost select \* from stock

SELECT Hash Match (Right Outer Join) Cost: 0 %

Index Scan (Clustered) [PK\_stock] Cost: 4 %

Index Scan (Clustered) [PK\_invoice] Cost: 5 %

ACER-VISHWA\Vishwa (73) inventory\_management\_s... 00:00:00 0 rows

## 5. SUB QUERIES – stock, invoice, vendor

SQLQuery1.sql - ACER-VISHWA\SQLEXPRESS01.inventory\_management\_system (ACER-VISHWA\Vishwa (73)) - Microsoft SQL Server Management Studio

```
File Edit View Query Project Tools Window Help
File Object Explorer Tools Database Diagrams
[+] ACER-VISHWA\SQLEXPRESS01 (SQL Server 11)
    Databases
        System Databases
        Database Snapshots
        inventory_management_system
        Database Diagrams
    Tables
        System Tables
        FileTables
        External Tables
        Graph Tables
        dbo.customer
        dbo.invoice
        dbo.login
        dbo.product
        dbo.stock
        dbo.vendor
    Views
    External Resources
    Synonyms
    Programmability
    Query Store
    Service Broker
    Storage
    Security
    Ready
```

Object Explorer

SQLQuery1.sql - A...JISHWA\Vis... Nested Loops (Inner Join)

```
select * from stock where stock.pid IN (select invoice.prod_id from invoice where invoice.prod_id IN (select vendor.product_id from vendor));
```

Physical Operation Nested Loops (Inner Join)

Logical Operation Sort (Distinct Sort)

Estimated Execution Mode Hash Match (Left Semi Join)

Estimated Operator Cost 0 %

Estimated I/O Cost 0 %

Estimated Subtree Cost 100 %

Estimated CPU Cost 0 %

Estimated Number of Executions 1

Estimated Number of Rows Per Execution 3795

Estimated Number of Rows for All Executions 3795

Estimated Row Size 230 B

Node ID 0

Output List

```
[inventory_management_system].[dbo].[stock].pid,
[inventory_management_system].[dbo].[stock].purchase_date,
[inventory_management_system].[dbo].[stock].mfg_date,
[inventory_management_system].[dbo].[stock].pqty,
[inventory_management_system].[dbo].[invoice].prod_id,
[inventory_management_system].[dbo].[invoice].invoice_no,
[inventory_management_system].[dbo].[invoice].invoice_date,
[inventory_management_system].[dbo].[invoice].selling_price,
[inventory_management_system].[dbo].[invoice].prod_id,
```

Hash Keys Probe

```
[inventory_management_system].[dbo].[vendor].product_id
```

Probe Residual

```
[inventory_management_system].[dbo].[invoice].prod_id =
[inventory_management_system].[dbo].[vendor].product_id]
```

Messages 1# Execution plan

Query 1: Query cost (relative to the batch): 100%

select \* from stock where stock.pid IN (select invoice.prod\_id from invoice where invoice.prod\_id IN (select vendor.product\_id from vendor));

SELECT Nested Loops (Inner Join) Cost: 0 %

Sort (Distinct Sort) Cost: 15 %

Hash Match (Left Semi Join) Cost: 46 %

Clustered Index Seek (Clustered) [stock].[PK\_stock] Cost: 5 %

Clustered Index Scan (Clustered) [invoice].[PK\_invoice] Cost: 7 %

Table Scan [vendor] Cost: 27 %

ACER-VISHWA\SQLEXPRESS01 (1...) ACER-VISHWA\Vishwa (73) inventory\_management\_s... 00:00:00 0 rows

SQLQuery1.sql - ACER-VISHWA\SQLEXPRESS01.inventory\_management\_system (ACER-VISHWA\Vishwa (73)) - Microsoft SQL Server Management Studio

```
File Edit View Query Project Tools Window Help
File Object Explorer Tools Status Bar
[ inventory_management_sy... ] Execute
Nested Loops
select * from stock where stock.pid IN (select invoice.prod_id from invoice where invoice.prod_id IN (select vendor.product_id from vendor));
Physical Operation
Nested Loops (Inner Join)
Logical Operation
Inner Join
Estimated Execution Mode Row
Estimated I/O Cost 0
Estimated CPU Cost 0.0000212 (0%)
Estimated Operator Cost 0.0000211
Estimated Subtree Cost 0.0754528
Estimated Number of Executions 1
Estimated Number of Rows Per Execution 255
Estimated Number of Rows for All Executions 255
Estimated Row Size 18 B
Node ID 1
Output List
[inventory_management_system].[dbo].[stock].pid,
[inventory_management_system].[dbo].[stock].purchase_date,
[inventory_management_system].[dbo].[stock].mfg_date,
[inventory_management_system].[dbo].[stock].qty
Outer References
[inventory_management_system].[dbo].[invoice].prod_id
[inventory_management_system].[dbo].[vendor].product_id
[ACER-VISHWA\Vishwa (73) inventory_management_s... 00:00:00 0 rows]
Ready
```

## 6. SUB QUERIES – stock, vendor, invoice

SQLQuery1.sql - ACER-VISHWA\SQLEXPRESS01.inventory\_management\_system (ACER-VISHWA\Vishwa (73)) - Microsoft SQL Server Management Studio

```
File Edit View Query Project Tools Window Help
File Object Explorer Tools Status Bar
[ inventory_management_sy... ] Execute
Nested Loops
select * from stock where stock.pid IN (select vendor.product_id from vendor where vendor.product_id IN (select invoice.prod_id from invoice));
Physical Operation
SELECT [Sort] [Distinct Sort] Hash Match [Inner Join] Nested Loops [Inner Join] Stream Aggregate [Aggregate] Clustered Index Scan [C... [invoice].[PK_invoice]
Cost: 0 % Cost: 15 % Cost: 47 % Cost: 0 % Cost: 0 % Cost: 6 %
Table Scan [vendor]
Cost: 26 %
Clustered Index Seek [C... [stock].[PK_stock]
Cost: 6 %
[ACER-VISHWA\SQLEXPRESS01 (1... | ACER-VISHWA\Vishwa (73) inventory_management_s... 00:00:00 0 rows]
Ready
```

The screenshot shows the Microsoft SQL Server Management Studio interface. The Object Explorer on the left lists the database structure for 'ACER-VISHWA\SQLEXPRESS01'. The 'Tables' node under 'inventory\_management\_system' contains several tables: stock, vendor, product, invoice, customer, login, and others. The 'Messages' tab in the center displays the following T-SQL query:

```
select * from stock where stock.pid IN (select vendor.product_id from vendor where vendor.product_id IN (select invoice.prod_id from invoice));
```

Below the query, the 'Execution plan' pane shows the estimated cost and details of the execution plan. The plan consists of two main operations:

- A **Stream Aggregate** operation with an **Aggregate** operator. It has a cost of 0 % and is part of a logical operation.
- A **Clustered Index Scan** operation on the **[invoice].[PK\_invoice]** index. It has a cost of 6 % and is part of a physical operation.

The execution plan also includes a **Clustered Index Seek** operation on the **[stock].[PK\_stock]** index, which has a cost of 6 %.

**Sort** statistics are provided for both the Stream Aggregate and Clustered Index Scan operations.

**Output List** and **Order By** clauses are also visible at the bottom of the execution plan pane.

## 7. LEFT OUTER JOIN – invoice, product, vendor

The screenshot shows the Microsoft SQL Server Management Studio interface. The title bar indicates the connection is to ACER-VISHWA\SQLEXPRESS01.inventory\_management\_system (ACER-VISHWA\Vishwa (73)) - Microsoft SQL Server Management Studio. The Object Explorer sidebar shows the database structure, including the inventory\_management\_system database. The main window displays a query in the SQL Query Editor:

```
select * from invoice LEFT OUTER JOIN product on invoice.prod_id = product.product_id  
LEFT OUTER JOIN vendor on invoice.prod_id = vendor.product_id;
```

The execution plan for this query is shown below the query text. It consists of three main operations:

- A **SELECT** operation (Cost: 0 %) which performs a **Hash Match (Left Outer Join)** with a cost of 65 %.
- A **Table Scan** operation (Cost: 15 %) for the **[vendor]** table, which is part of a **Hash Match (Right Outer Join)** with a cost of 15 %.
- A **Clustered Index Scan** operation (Cost: 2 %) for the **[product]** table, which is part of a **Clustered Index Scan (C...** operation with a cost of 4 %.

The overall query cost is 100 %.

SQLQuery1.sql - ACER-VISHWA\SQLEXPRESS01.inventory\_management\_system (ACER-VISHWA\Vishwa (73)) - Microsoft SQL Server Management Studio

```
select * from
[LEFT OUTER JOIN]
product.product_id
```

**Hash Match**  
Use each row from the top input to build a hash table, and each row from the bottom input to probe into the hash table, outputting all matching rows.

Physical Operation	Hash Match
Logical Operation	Left Outer Join
Estimated Execution Mode	Row
Estimated Operator Cost	0.0898046 (65%)
Estimated I/O Cost	0
Estimated Subtree Cost	0.138607
Estimated CPU Cost	0.0898016
Estimated Number of Executions	1
Estimated Number of Rows Per Execution	2026
Estimated Number of Rows for All Executions	2026
Estimated Row Size	331 B
Node ID	0

**Output List**

```
[inventory_management_system].[dbo].[invoice].prod_id,
[inventory_management_system].[dbo].[invoice].invoice_no,
[inventory_management_system].[dbo].[invoice].invoice_date,
[inventory_management_system].[dbo].[invoice].selling_price,
[inventory_management_system].[dbo].[invoice].prod_qty,
[inventory_management_system].[dbo].[invoice].discount,
[inventory_management_system].[dbo].[invoice].phone_no,
[inventory_management_system].[dbo].[invoice].vendor_id,
[inventory_management_system].[dbo].[product].p...
```

**Hash Keys Probe**

```
[inventory_management_system].[dbo].[vendor].product_id
```

**Probe Residual**

```
[inventory_management_system].[dbo].[invoice].prod_id =
[inventory_management_system].[dbo].[vendor].product_id
```

**INS**

## 8. LEFT OUTER JOIN – vendor, invoice, product

SQLQuery1.sql - ACER-VISHWA\SQLEXPRESS01.inventory\_management\_system (ACER-VISHWA\Vishwa (73)) - Microsoft SQL Server Management Studio

```
select * from vendor LEFT OUTER JOIN invoice on invoice.prod_id = vendor.product_id
[LEFT OUTER JOIN] product on product.product_id = invoice.prod_id;
```

**Query cost (relative to the batch): 100%**

**Execution plan:**

```
graph TD
    A[SELECT ...] --> B[Hash Match (Right Outer Join)]
    B --> C[Clustered Index Scan ... [product].[PK_product]]
    C --> D[Hash Match (Right Outer Join)]
    D --> E[Clustered Index Scan ... [invoice].[PK_invoice]]
    E --> F[Table Scan ... [vendor]]
```

**INS**

SQLQuery1.sql - ACER-VISHWA\SQLEXPRESS01.inventory\_management\_system (ACER-VISHWA\Vishwa (73)) - Microsoft SQL Server Management Studio

```
File Edit View Query Project Tools Window Help
File Connect New Query File New Object Explorer
inventory_management_sy... Execute
Object Explorer
Connect > inventory_management_sy...
ACER-VISHWA\SQLEXPRESS01 (SQL Server 11)
Databases System Databases Database Snapshots inventory_management_system Database Diagrams Tables System Tables FileTables External Tables Graph Tables dbo.customer dbo.invoice dbo.login dbo.product dbo.stock dbo.vendor Views External Resources Synonyms Programmability Query Store Service Broker Storage Security
Ready
SQLQuery1.sql - A...ISHWA\...
select * from vendor
LEFT OUTER JOIN
Hash Match
Use each row from the top input to build a hash table, and each row from the bottom input to probe into the hash table, outputting all matching rows.

Physical Operation Hash Match
Logical Operation Right Outer Join
Estimated Execution Mode Row
Estimated Operator Cost 0.0529428 (42%)
Estimated I/O Cost 0
Estimated Subtree Cost 0.126954
Estimated CPU Cost 0.0529401
Estimated Number of Executions 1
Estimated Number of Rows Per Execution 7591.96
Estimated Number of Rows for All Executions 7591.96
Estimated Row Size 331.8
Node ID 0

Output List
[inventory_management_system].[dbo].[vendor].fname,
[inventory_management_system].[dbo].[vendor].lname,
[inventory_management_system].[dbo].[vendor].vendor_id,
[inventory_management_system].[dbo].[vendor].email_id,
[inventory_management_system].[dbo].[vendor].product_id,
[inventory_management_system].[dbo].[vendor].cust_id,
[inventory_management_system].[dbo].[invoice].prod_id,
[inventory_management_system].[dbo].[invoice].invoice_no,
[inventory_management_system].[dbo].[invoice].invoice_date...
Hash Keys Probe
[inventory_management_system].[dbo].[invoice].prod_id
[inventory_management_system].[dbo].[product].[product_id]=
[inventory_management_system].[dbo].[invoice].[prod_id]

Probe Residual
[ACER-VISHWA\Vishwa (73) inventory_management_s... 00:00:00 0 rows]
NS

```

## 9. RIGHT OUTER JOIN – vendor, invoice, product

SQLQuery1.sql - ACER-VISHWA\SQLEXPRESS01.inventory\_management\_system (ACER-VISHWA\Vishwa (73)) - Microsoft SQL Server Management Studio

```
File Edit View Query Project Tools Window Help
File Connect New Query File New Object Explorer
inventory_management_sy... Execute
Object Explorer
Connect > inventory_management_sy...
ACER-VISHWA\SQLEXPRESS01 (SQL Server 11)
Databases System Databases Database Snapshots inventory_management_system Database Diagrams Tables System Tables FileTables External Tables Graph Tables dbo.customer dbo.invoice dbo.login dbo.product dbo.stock dbo.vendor Views External Resources Synonyms Programmability Query Store Service Broker Storage Security
Ready
SQLQuery1.sql - A...ISHWA\Vishwa (73)* ...
select * from vendor RIGHT OUTER JOIN invoice on invoice.prod_id = vendor.product_id;
RIGHT OUTER JOIN product on invoice.prod_id = product.product_id;

100 %
Messages 0% Execution plan
Query 1: Query cost (relative to the batch): 100%
select * from vendor RIGHT OUTER JOIN invoice on invoice.prod_id = vendor.product_id RIGHT OUTER JOIN product on invoice.prod_id = product.product_id;

SELECT Hash Match (Left Outer Join) Cost: 0 %
[product].[PK_product]
Cost: 29 %
Clustered Index Scan (C...
[product].[PK_product]
Cost: 3 %

Hash Match (Left Outer Join) Cost: 43 %
[invoice].[PK_invoice]
Cost: 5 %

Table Scan [vendor]
Cost: 20 %

[ACER-VISHWA\SQLEXPRESS01 (1... ACER-VISHWA\Vishwa (73) inventory_management_s... 00:00:00 0 rows]
NS

```

SQLQuery1.sql - ACER-VISHWA\SQLEXPRESS01.inventory\_management\_system (ACER-VISHWA\Vishwa (73)) - Microsoft SQL Server Management Studio

```
File Edit View Query Project Tools Window Help
New Query Execute
Object Explorer
Connect ▾ inventory_management_system
ACER-VISHWA\SQLEXPRESS01 (SQL Server 11)
Databases System Databases Database Snapshots Database Diagrams Tables System Tables FileTables External Tables Graph Tables dbo.customer dbo.invoice dbo.login dbo.product dbo.stock dbo.vendor Views External Resources Synonyms Programmability Query Store Service Broker Storage Security
Ready
inventory_management_sy... Execute
SQLQuery1.sql - A...ISHWA\Vishwa (73) * X
select * from vendor RIGHT OUTER JOIN invoice on invoice.prod_id = vendor.product_id;
RIGHT OUTER JOIN product on invoice.prod_id = product.product_id;
```

Hash Match  
Use each row from the top input to build a hash table, and each row from the bottom input to probe into the hash table, outputting all matching rows.

**Physical Operation**

Logical Operation	Hash Match
Estimated Execution Mode	Left Outer Join
Estimated Operator Cost	0.0293615 (29%)
Estimated I/O Cost	0
Estimated Subtree Cost	0.101003
Estimated CPU Cost	0.0293583
Estimated Number of Executions	1
Estimated Number of Rows Per Execution	2026
Estimated Number of Rows for All Executions	2026
Estimated Row Size	331 B
Node ID	0

**Output List**

- [inventory\_management\_system].[dbo].[vendor].fname,
- [inventory\_management\_system].[dbo].[vendor].lname,
- [inventory\_management\_system].[dbo].[vendor].vendor\_id,
- [inventory\_management\_system].[dbo].[vendor].email\_id,
- [inventory\_management\_system].[dbo].[vendor].product\_id,
- [inventory\_management\_system].[dbo].[vendor].cust\_id,
- [inventory\_management\_system].[dbo].[invoice].prod\_id,
- [inventory\_management\_system].[dbo].[invoice].invoice\_no,
- [inventory\_management\_system].[dbo].[invoice].invoice\_date...

Hash Keys Probe [inventory\_management\_system].[dbo].[invoice].prod\_id

Messages 0 Execution plan

Query 1: Query cost (relative to the batch): 100%

select \* from invoice RIGHT OUTER JOIN product on invoice.prod\_id = product.product\_id RIGHT OUTER JOIN vendor on invoice.prod\_id = vendor.product\_id;

Table Scan [vendor] Cost: 20 %

Table Scan [product] Cost: 3 %

Clustered Index Seek ([invoice].[PK\_invoice]) Cost: 12 %

Clustered Index Scan ([product].[PK\_product]) Cost: 3 %

Nested Loops (Inner Join) Cost: 1 %

Hash Match (Right Outer Join) Cost: 64 %

SELECT Cost: 0 %

Query executed successfully!

ACER-VISHWA\SQLEXPRESS01 (1...) ACER-VISHWA\Vishwa (73) inventory\_management\_s... 00:00:00 0 rows

## 10. RIGHT OUTER JOIN – invoice, product, vendor

SQLQuery1.sql - ACER-VISHWA\SQLEXPRESS01.inventory\_management\_system (ACER-VISHWA\Vishwa (73)) - Microsoft SQL Server Management Studio

```
File Edit View Query Project Tools Window Help
New Query Execute
Object Explorer
Connect ▾ inventory_management_system
ACER-VISHWA\SQLEXPRESS01 (SQL Server 11)
Databases System Databases Database Snapshots Database Diagrams Tables System Tables FileTables External Tables Graph Tables dbo.customer dbo.invoice dbo.login dbo.product dbo.stock dbo.vendor Views External Resources Synonyms Programmability Query Store Service Broker Storage Security
Ready
inventory_management_sy... Execute
SQLQuery1.sql - A...ISHWA\Vishwa (73) * X
select * from invoice RIGHT OUTER JOIN product on invoice.prod_id = product.product_id;
RIGHT OUTER JOIN vendor on invoice.prod_id = vendor.product_id;
```

Messages 0 Execution plan

Query 1: Query cost (relative to the batch): 100%

select \* from invoice RIGHT OUTER JOIN product on invoice.prod\_id = product.product\_id RIGHT OUTER JOIN vendor on invoice.prod\_id = vendor.product\_id;

Table Scan [vendor] Cost: 20 %

Table Scan [product] Cost: 3 %

Clustered Index Seek ([invoice].[PK\_invoice]) Cost: 12 %

Clustered Index Scan ([product].[PK\_product]) Cost: 3 %

Nested Loops (Inner Join) Cost: 1 %

Hash Match (Right Outer Join) Cost: 64 %

SELECT Cost: 0 %

Query executed successfully!

ACER-VISHWA\SQLEXPRESS01 (1...) ACER-VISHWA\Vishwa (73) inventory\_management\_s... 00:00:00 0 rows

The screenshot shows the Microsoft SQL Server Management Studio interface. The Object Explorer on the left lists the database structure for 'ACER-VISHWA\SQLEXPRESS01'. The main pane displays a query window titled 'SQLQuery1.sql - A\_ISHWA\Vishwa' containing the following T-SQL code:

```
select * from invoice
RIGHT OUTER JOIN vendor
ON invoice.vendor_id = vendor.vendor_id
```

An execution plan is shown for this query, highlighting a 'Hash Match' operation. The plan details are as follows:

Physical Operation		Hash Match
Logical Operation		Right Outer Join
Estimated Execution Mode		Row
Estimated Operator Cost	0.0645959 (64%)	
Estimated I/O Cost	0	
Estimated Subtree Cost	0.100913	
Estimated CPU Cost	0.0645926	
Estimated Number of Executions	1	
Estimated Number of Rows Per Execution	3796	
Estimated Number of Rows for All Executions	3796	
Estimated Row Size	331 B	
Node ID	0	

The output list for the query includes columns from both the 'invoice' and 'vendor' tables. The 'Ready' status bar at the bottom indicates the query was executed successfully.

## 11. SUB QUERIES – product, invoice, vendor

The screenshot shows the Microsoft SQL Server Management Studio interface. The Object Explorer on the left lists the database structure for 'ACER-VISHWA\SQLEXPRESS01'. The 'Tables' node under 'inventory\_management\_system' contains several tables: dbo.customer, dbo.invoice, dbo.login, dbo.product, dbo.stock, and dbo.vendor. The 'Messages' tab in the center shows a query execution plan for the following T-SQL code:

```
select * from product where product.product_id IN (select invoice.prod_id from invoice where invoice.prod_id IN (select vendor.product_id from vendor));
```

The execution plan details the cost of each operation:

- SELECT**: Cost: 0 %
- Nested Loops (Inner Join)**: Cost: 0 %
- Sort (Distinct Sort)**: Cost: 15 %
- Hash Match (Left Semi Join)**: Cost: 46 %
- Clustered Index Scan ([invoice].[PK\_invoice])**: Cost: 7 %
- Clustered Index Seek ([product].[PK\_product])**: Cost: 5 %
- Table Scan [vendor]**: Cost: 27 %

At the bottom, a green status bar indicates: "Query executed successfully." The status bar also shows the connection information: ACER-VISHWA\SQLEXPRESS01 (1...) ACER-VISHWA\Vishwa (73) inventory\_management\_s... 00:00:00 0 rows.

SQLQuery1.sql - ACER-VISHWA\SQLEXPRESS01.inventory\_management\_system (ACER-VISHWA\Vishwa (73)) - Microsoft SQL Server Management Studio

```
File Edit View Query Project Tools Window Help
File Connect New Query Execute
inventory_management_sy... Quick Launch (Ctrl+Q) X
Object Explorer
Connect > inventory_management_sy...
ACER-VISHWA\SQLEXPRESS01 (SQL Server 11)
Databases
System Databases
Database Snapshots
inventory_management_system
Database Diagrams
Tables
System Tables
FileTables
External Tables
Graph Tables
dbo.customer
dbo.invoice
dbo.login
dbo.product
dbo.stock
dbo.vendor
Views
External Resources
Synonyms
Programmability
Query Store
Service Broker
Storage
Security
Matches: (
```

SQLQuery1.sql - A...ISHWA\Vishwa (73)\*

```
select * from product where product.product_id IN (select invoice.prod_id from invoice where invoice.prod_id IN (select vendor.product_id from vendor));
```

Nested Loops

Physical Operation

SELECT Cost: 0 %

Nested Loops (Inner Join)

Estimated Execution Mode: Row

Estimated I/O Cost: 0

Estimated Operator Cost: 0.0000212 (0%)

Estimated CPU Cost: 0.0000211

Estimated Subtree Cost: 0.0754528

Estimated Number of Executions: 1

Estimated Number of Rows Per Execution: 13

Estimated Number of Rows for All Executions: 13

Estimated Row Size: 121 B

Node ID: 1

Output List:

- [inventory\_management\_system].[dbo].[product].product\_id,
- [inventory\_management\_system].[dbo].[product].product\_name,
- [inventory\_management\_system].[dbo].[product].cost\_price,
- [inventory\_management\_system].[dbo].[product].manufacturer,
- [inventory\_management\_system].[dbo].[product].mrp

Outer References:

- [inventory\_management\_system].[dbo].[invoice].prod\_id

ACER-VISHWA\Vishwa (73) inventory\_management\_s... 00:00:00 0 rows

INS

## 12. SUB QUERIES – product, vendor, invoice

SQLQuery1.sql - ACER-VISHWA\SQLEXPRESS01.inventory\_management\_system (ACER-VISHWA\Vishwa (73)) - Microsoft SQL Server Management Studio

```
File Edit View Query Project Tools Window Help
File Connect New Query Execute
inventory_management_sy... Quick Launch (Ctrl+Q) X
Object Explorer
Connect > inventory_management_sy...
ACER-VISHWA\SQLEXPRESS01 (SQL Server 11)
Databases
System Databases
Database Snapshots
inventory_management_system
Database Diagrams
Tables
System Tables
FileTables
External Tables
Graph Tables
dbo.customer
dbo.invoice
dbo.login
dbo.product
dbo.stock
dbo.vendor
Views
External Resources
Synonyms
Programmability
Query Store
Service Broker
Storage
Security
Matches: (
```

SQLQuery1.sql - A...ISHWA\Vishwa (73)\*

```
select * from product where product.product_id IN (select vendor.product_id from vendor where vendor.product_id IN (select invoice.prod_id from invoice));
```

100 %

Execution plan

Query 1: Query cost (relative to the batch): 100%

select \* from product where product.product\_id IN (select vendor.product\_id from vendor where vendor.product\_id IN (select invoice.prod\_id from invoice));

Physical Operation

SELECT Cost: 0 %

Sort (Distinct Sort) Cost: 15 %

Hash Match (Inner Join) Cost: 44 %

Nested Loops (Inner Join) Cost: 0 %

Stream Aggregate (Aggregate) Cost: 0 %

Clustered Index Scan ([invoice].[PK\_invoice]) Cost: 7 %

Table Scan [vendor] Cost: 28 %

Clustered Index Seek ([product].[PK\_product]) Cost: 6 %

ACER-VISHWA\SQLEXPRESS01 (1...) ACER-VISHWA\Vishwa (73) inventory\_management\_s... 00:00:00 0 rows

Ready

The screenshot shows the Microsoft SQL Server Management Studio interface. In the Object Explorer, the database 'inventory\_management\_system' is selected. In the center pane, a query window displays the following T-SQL code:

```
select * from product where product.product_id IN (select vendor.product_id from vendor where vendor.product_id IN (select invoice.prod_id from invoice));
```

Below the code, the 'Execution Plan' window is open, showing the estimated cost and execution details. The estimated cost is 0.0130995. The execution plan diagram illustrates the join operations between the three tables.

**Comparison of estimated subtree costs for different join operations and indexes:**

Operation	Estimated Subtree Cost
1. LEFT OUTER - invoice, stock, vendor	0.0965849
2. LEFT OUTER (Order Changed) - vendor, invoice, stock	0.0130995
3. RIGHT OUTER - vendor, invoice, stock	0.10393
4. RIGHT OUTER (Order Changed) - invoice, stock, vendor	0.0989511
5. SUB QUERIES - stock, invoice, vendor	0.0754528
6. SUB QUERIES (Order Changed) - stock, vendor, invoice	0.0785116
7. LEFT OUTER - invoice, product, vendor	0.08988016
8. LEFT OUTER (Order Changed) - vendor, invoice, product	0.0529401
9. RIGHT OUTER - vendor, invoice, product	0.0293583
10. RIGHT OUTER (Order Changed) - invoice, product, vendor	0.06455926
11. SUB QUERIES - product, invoice, vendor	0.0754528
12. SUB QUERIES (Order Changed) - product, vendor, invoice	0.0736339