



Database Concepts Report with Practical Examples

Submitted by: Sama Wael Abdou

Track: Data Engineering



Scenario 1 – Synonyms

Definition

A synonym is an alternative name given to an existing database object. It does not create a copy, just a shortcut.

Purpose

- Shortens long object names
- Makes queries easier to read
- Can point to objects in another schema or database
- Helps in maintaining large systems

Example:

```
58  
59  
60 CREATE SYNONYM Std FOR Students;  
61  
62 SELECT * FROM Std;
```

Explanation of Output:

The query returns all rows from Students.

Scenario 2 – Window Functions

Definition

Window functions allow calculations across a set of rows while keeping the original rows in the result set.

Purpose

- Rank rows without reducing data
- Compute running totals, moving averages, and ranks per group

Common Window Functions

- Row_Number()
- Rank()
- Dense_Rank()



Example:

```
63
64 SELECT
65     CourseID,
66     StudentID,
67     Grade,
68     RANK() OVER (PARTITION BY CourseID ORDER BY Grade DESC) AS RankInCourse
69 FROM Enrollments;
```

00 % No issues found

CourseID	StudentID	Grade	RankInCourse
101	1	85.00	1
101	3	78.00	2
102	2	90.00	1
103	4	88.00	1

Explanation of Output:

Each student is ranked within their course, all rows remain visible.

Scenario 3 – Advanced Grouping Techniques

3.1 ROLLUP

- **Definition:** Creates subtotals in a hierarchical order.

Example:

```
63
64 SELECT
65     CourseID,
66     StudentID,
67     Grade,
68     RANK() OVER (PARTITION BY CourseID ORDER BY Grade DESC) AS RankInCourse
69 FROM Enrollments;
```

00 % No issues found

CourseID	StudentID	Grade	RankInCourse
101	1	85.00	1
101	3	78.00	2
102	2	90.00	1
103	4	88.00	1

Explanation of Output:

Shows total grades per course and a grand total.



3.2 CUBE

- **Definition:** Generates all possible subtotal combinations.

Example:

```
75
76
77 SELECT CourseID, StudentID, SUM(Grade) AS TotalGrade
78 FROM Enrollments
79 GROUP BY CUBE(CourseID, StudentID);
```

100 % No issues found

	CourseID	StudentID	TotalGrade
1	101	1	85.00
2	NULL	1	85.00
3	102	2	90.00
4	NULL	2	90.00
5	101	3	78.00
6	NULL	3	78.00
7	103	4	88.00
8	NULL	4	88.00
9	NULL	NULL	341.00
10	101	NULL	163.00
11	102	NULL	90.00
12	103	NULL	88.00

Explanation of Output:

Includes all combinations of totals per course, per student, and grand total.

3.3 GROUPING SETS

- **Definition:** Define exactly which summaries you want in a single query.

Example:

```
81
82 SELECT CourseID, StudentID, SUM(Grade) AS TotalGrade
83 FROM Enrollments
84 GROUP BY GROUPING SETS(
85     (CourseID),
86     (StudentID),
87     ()
88 );
```

100 % No issues found

	CourseID	StudentID	TotalGrade
1	NULL	1	85.00
2	NULL	2	90.00
3	NULL	3	78.00
4	NULL	4	88.00
5	NULL	NULL	341.00
6	101	NULL	163.00
7	102	NULL	90.00
8	103	NULL	88.00

Explanation of Output:

Returns total per course, per student, and grand total in one query.

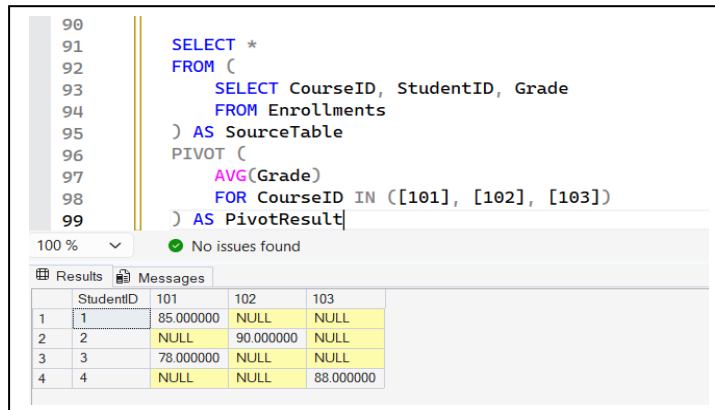


Scenario 4 – Pivoting

Definition

Pivoting converts row values into columns, useful in reporting.

Example:



```
90
91 SELECT *
92 FROM (
93     SELECT CourseID, StudentID, Grade
94     FROM Enrollments
95 ) AS SourceTable
96 PIVOT (
97     AVG(Grade)
98     FOR CourseID IN ([101], [102], [103])
99 ) AS PivotResult
```

100 % No issues found

	StudentID	101	102	103
1	1	85.000000	NULL	NULL
2	2	NULL	90.000000	NULL
3	3	78.000000	NULL	NULL
4	4	NULL	NULL	88.000000

Explanation of Output:

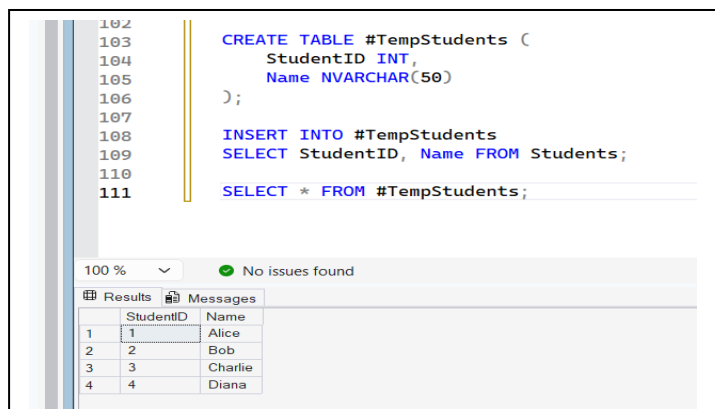
Each row is a student; columns show average grade per course.

Scenario 5 – Local and Global Temporary Tables

Local Temporary Table

- **Definition:** Exists only in current session.
- **Scope:** Current connection only.
- **Lifetime:** Deleted when session ends.

Example:



```
102
103 CREATE TABLE #TempStudents (
104     StudentID INT,
105     Name NVARCHAR(50)
106 );
107
108 INSERT INTO #TempStudents
109 SELECT StudentID, Name FROM Students;
110
111 SELECT * FROM #TempStudents;
```

100 % No issues found

	StudentID	Name
1	1	Alice
2	2	Bob
3	3	Charlie
4	4	Diana



Global Temporary Table

- **Definition:** Accessible across sessions.
- **Lifetime:** Deleted when the last session ends.

Example:

```
113
114 CREATE TABLE ##GlobalStudents (
115     StudentID INT,
116     Name NVARCHAR(50)
117 );
118
119 INSERT INTO ##GlobalStudents
120 SELECT StudentID, Name FROM Students;
121
122 SELECT * FROM ##GlobalStudents;
```

100 % No issues found

	StudentID	Name
1	1	Alice
2	2	Bob
3	3	Charlie
4	4	Diana

Difference Table:

Feature	Local Temp	Global Temp
Prefix	#	##
Scope	Current session	All sessions
Lifetime	Session end	Last session ends

Scenario 6 – Database Backup Types

Full Backup

Example:

```
124
125 BACKUP DATABASE DatabaseTask
126 TO DISK = 'D:\ITI Data Engineering\ITI - DataEngineering - journey\Database\Database Report\DatabaseTask_Full.bak'
127 WITH FORMAT,
128 NAME = 'Full Backup of DatabaseTask';
```

100 % No issues found Ln: 125, Ch: 1 (197 chars, 4 lines) SPC CRLF Wind

Messages

Processed 584 pages for database 'DatabaseTask', file 'DatabaseTask' on file 1.
Processed 2 pages for database 'DatabaseTask', file 'DatabaseTask_log' on file 1.
BACKUP DATABASE successfully processed 586 pages in 0.034 seconds (134.535 MB/sec).

Completion time: 2026-02-13T01:37:45.0157144+02:00

Explanation: Full backup of the database



Differential Backup

Example:

```
129  
130 BACKUP DATABASE DatabaseTask  
131 TO DISK = 'D:\ITI Data Engineering\ITI - DataEngineering - journey\Database\Database Report\DatabaseTask_Diff.bak'  
132 WITH DIFFERENTIAL,  
133 NAME = 'Differential Backup of DatabaseTask'
```

Explanation: Back up only changes since last full backup.

Transaction Log Backup

Example:

```
134  
135 BACKUP LOG DatabaseTask  
136 TO DISK = 'D:\ITI Data Engineering\ITI - DataEngineering - journey\Database\Database Report\DatabaseTask_Log.trn'  
137 WITH NAME = 'Transaction Log Backup of DatabaseTask';
```

100 % No issues found Ln: 137, Ch: 54 SPC CRLF Win

Explanation: Allows point-in-time recovery.

Scenario 7 – Database Backup Strategy

Why Backups Are Critical

- Protect data from hardware failure, deletion, or crashes.

Strategy Example

- Daily full backup.
- Differential backup every 6 hours.
- Transaction log backup every 30 minutes.

Restore Scenario

1. Restore latest full backup.
2. Restore latest differential backup.
3. Restore transaction logs sequentially.

