

SQL CODING CHALLENGE

- 1) A) Execute OVER and PARTITION BY Clause in SQL Queries
- B) creating subtotals
- C) Total Aggregations using SQL Queries.

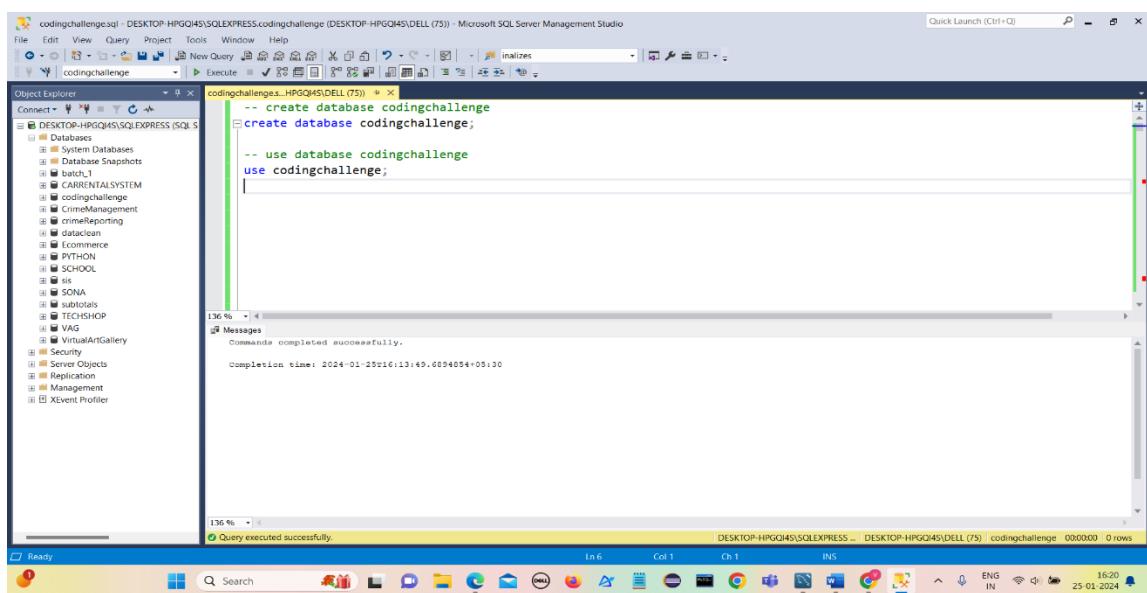
Creating a database:

Create a database named codingchallenge

Use the database:

`use codingchallenge;`

Use the database to make it as the current working database on which different operations can be performed.



The screenshot shows the Microsoft SQL Server Management Studio interface. In the Object Explorer on the left, there is a tree view of databases, including 'codingchallenge'. In the center pane, a query window titled 'codingchallenge.dbo.HRGQ4S(DELL (75))' contains the following SQL code:

```
-- create database codingchallenge;
create database codingchallenge;

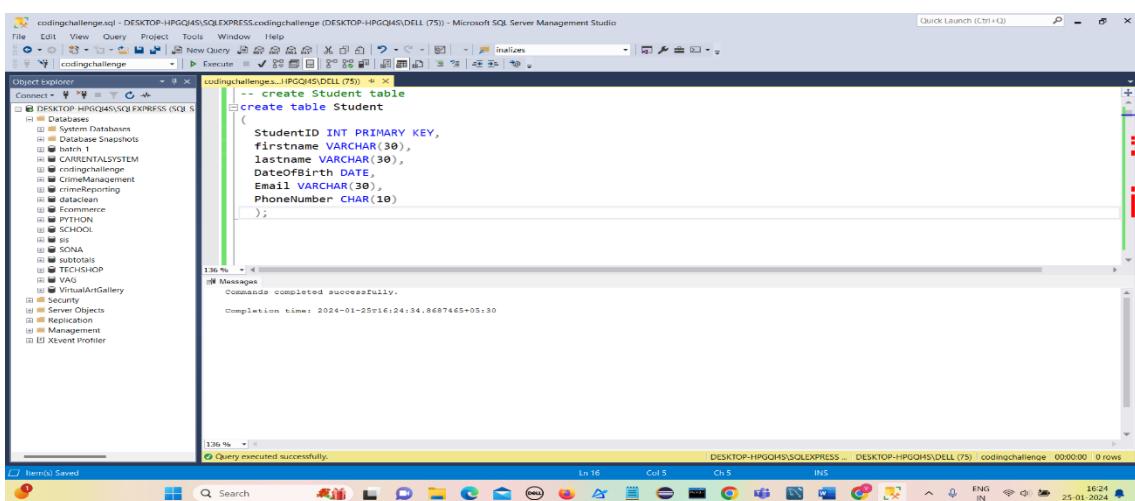
-- use database codingchallenge
use codingchallenge;
```

The 'Messages' pane at the bottom of the query window shows the output of the command:

```
Commands completed successfully.  
Completion time: 2024-01-25T16:13:49.6094054+05:30
```

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

1) Create student table using create statement and give attributes



The screenshot shows the Microsoft SQL Server Management Studio interface. In the Object Explorer on the left, there is a tree view of databases, including 'codingchallenge'. In the center pane, a query window titled 'codingchallenge.dbo.HRGQ4S(DELL (75))' contains the following SQL code:

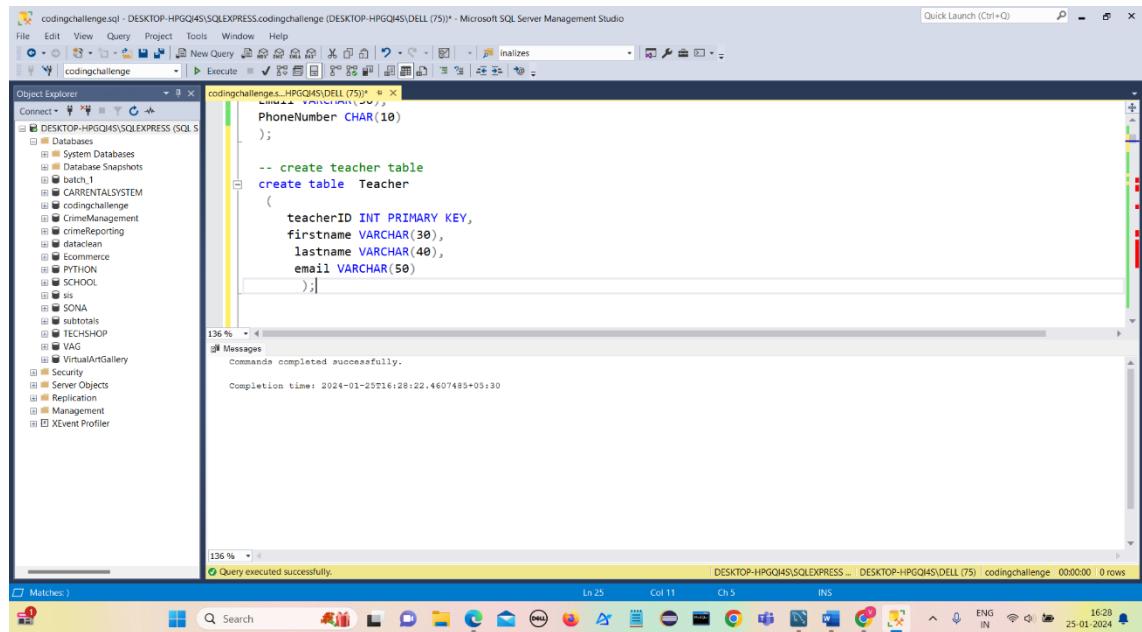
```
-- create Student table
create table Student
(
    StudentID INT PRIMARY KEY,
    firstname VARCHAR(30),
    lastname VARCHAR(30),
    DateOfBirth DATE,
    Email VARCHAR(30),
    PhoneNumber CHAR(10)
);
```

The 'Messages' pane at the bottom of the query window shows the output of the command:

```
Commands completed successfully.  
Completion time: 2024-01-25T16:24:34.8687465+05:30
```

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

2) Create table Teacher

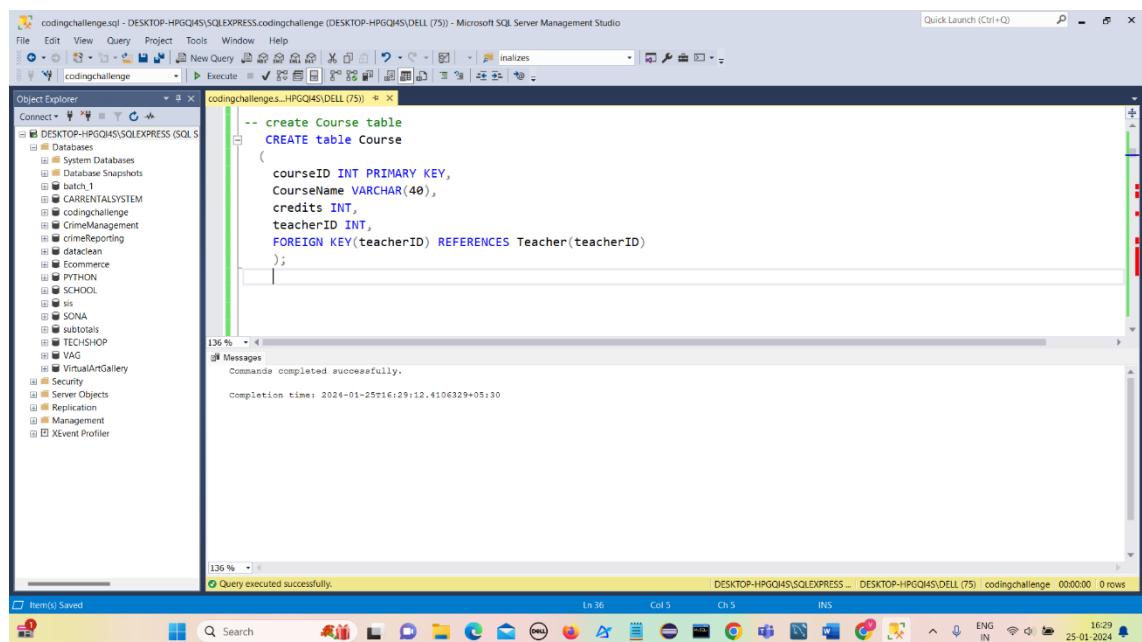


The screenshot shows the Microsoft SQL Server Management Studio interface. In the Object Explorer, a database named 'codingchallenge' is selected. In the main query window, the following SQL code is executed:

```
-- create teacher table
CREATE TABLE Teacher
(
    teacherID INT PRIMARY KEY,
    firstname VARCHAR(30),
    lastname VARCHAR(40),
    email VARCHAR(50)
);
```

The execution message in the status bar indicates: "Query executed successfully." and "Completion time: 2024-01-25T16:28:22.4607485+05:30".

3) Create Course Table

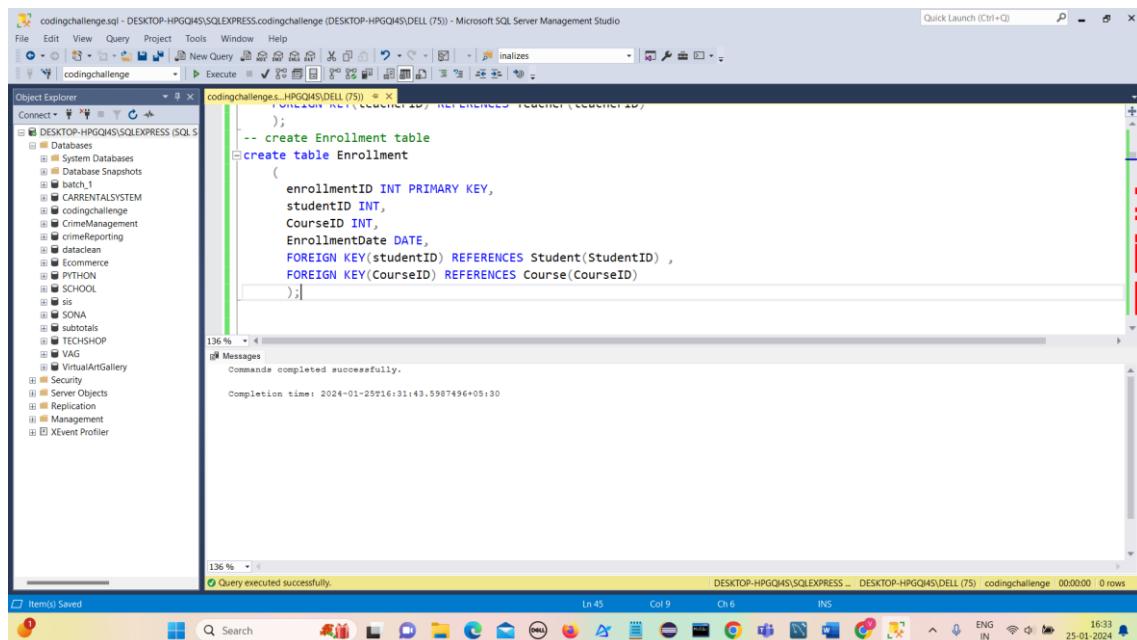


The screenshot shows the Microsoft SQL Server Management Studio interface. In the Object Explorer, a database named 'codingchallenge' is selected. In the main query window, the following SQL code is executed:

```
-- create Course table
CREATE TABLE Course
(
    courseID INT PRIMARY KEY,
    CourseName VARCHAR(40),
    credits INT,
    teacherID INT,
    FOREIGN KEY(teacherID) REFERENCES Teacher(teacherID)
);
```

The execution message in the status bar indicates: "Query executed successfully." and "Completion time: 2024-01-25T16:29:12.4106329+05:30".

4) Create Enrollment Table

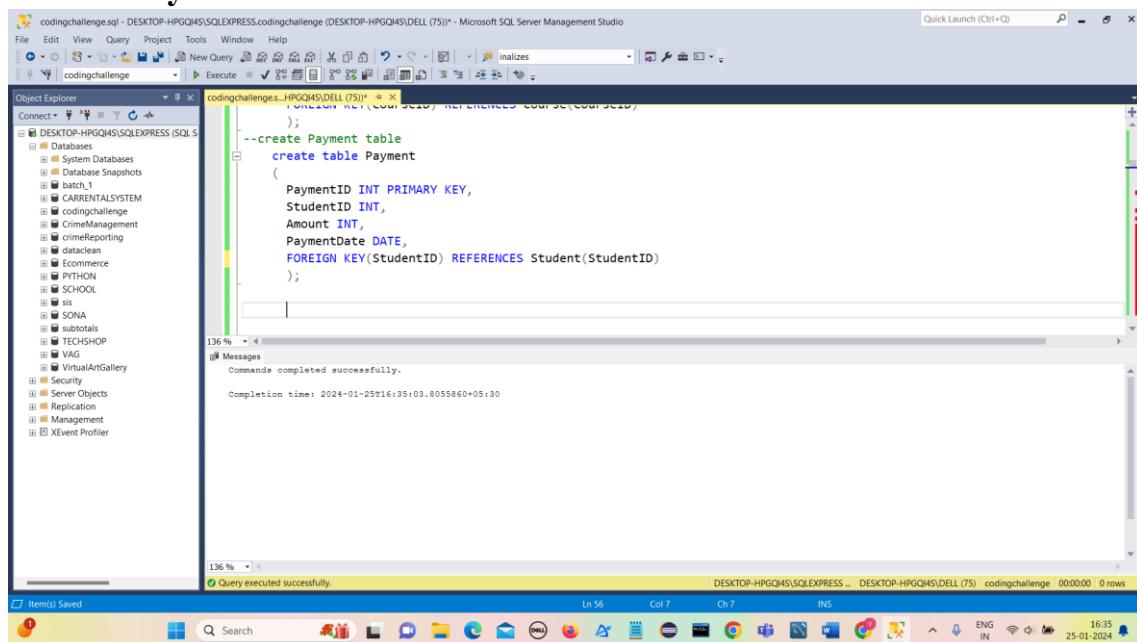


The screenshot shows the Microsoft SQL Server Management Studio interface. In the Object Explorer, a database named 'codingchallenge' is selected. In the main query window, the following SQL code is executed:

```
-- create Enrollment table
create table Enrollment
(
    enrollmentID INT PRIMARY KEY,
    studentID INT,
    CourseID INT,
    EnrollmentDate DATE,
    FOREIGN KEY(studentID) REFERENCES Student(StudentID),
    FOREIGN KEY(CourseID) REFERENCES Course(CourseID)
);
```

The execution completed successfully with the message "Commands completed successfully." and a completion time of 2024-01-25T16:31:43.5987496+05:30.

5) Create Payment table



The screenshot shows the Microsoft SQL Server Management Studio interface. In the Object Explorer, a database named 'codingchallenge' is selected. In the main query window, the following SQL code is executed:

```
--create Payment table
create table Payment
(
    PaymentID INT PRIMARY KEY,
    StudentID INT,
    Amount INT,
    PaymentDate DATE,
    FOREIGN KEY(StudentID) REFERENCES Student(StudentID)
);
```

The execution completed successfully with the message "Commands completed successfully." and a completion time of 2024-01-25T16:35:03.8055860+05:30.

6) Insert records into Student table using insert statement

The screenshot shows the Microsoft SQL Server Management Studio interface. In the Object Explorer, the database 'codingchallenge' is selected. In the main query window, an 'insert' statement is being run:

```
-- insert records into student table
insert into Student(StudentID,firstname,lastname,DateOfBirth,Email,PhoneNumber)
VALUES(101,'Vishnavi','Bacchu','2000-09-17','vish@gmail.com','9876543210'),
(102,'Avanthi','Mishra','2001-06-27','avanti@gmail.com','9800043210'),
(103,'Ashraf','Syed','2003-08-12','syed@gmail.com','9811143210'),
(104,'Nithin','Soma','2007-06-02','soma@gmail.com','9876500000'),
(105,'Maaya','Maaru','2000-01-01','maaya@gmail.com','8247238787'),
(106,'Rani','Vulture','2000-11-13','rani@gmail.com','9912247371'),
(107,'Archana','Puppala','2000-06-16','archana@gmail.com','9182320242'),
(108,'Snigha','Vole','2000-09-19','vole@gmail.com','9177364130'),
(109,'Sandhya','Vara','2000-06-11','vara@gmail.com','9701168624'),
(110,'Aryan','Sai','2000-12-19','sai@gmail.com','9870011122');
```

The status bar at the bottom indicates 'Completion time: 2024-01-25T16:37:05.2478143+05:30'.

Retrieve records from Student table using select statement

select * from Student;

The screenshot shows the Microsoft SQL Server Management Studio interface. In the Object Explorer, the database 'codingchallenge' is selected. In the main query window, a 'select' statement is being run:

```
-- retrieving records of student table
select * from student;
```

The results pane displays 10 rows of data:

StudentID	firstname	lastname	DateOfBirth	Email	PhoneNumber
101	Vishnavi	Bacchu	2000-09-17	vish@gmail.com	9876543210
102	Avanthi	Mishra	2001-06-27	avanti@gmail.com	9800043210
103	Ashraf	Syed	2003-08-12	syed@gmail.com	9811143210
104	Nithin	Soma	2007-06-02	soma@gmail.com	9876500000
105	Maaya	Maaru	2000-01-01	maaya@gmail.com	8247238787
106	Rani	Vulture	2000-11-13	rani@gmail.com	9912247371
107	Archana	Puppala	2000-06-16	archana@gmail.com	9182320242
108	Snigha	Vole	2000-09-19	vole@gmail.com	9177364130
109	Sandhya	Vara	2000-06-11	vara@gmail.com	9701168624
110	Aryan	Sai	2000-12-19	sai@gmail.com	9870011122

The status bar at the bottom indicates 'Completion time: 2024-01-25T16:37:05.2478143+05:30'.

7) Insert records into teacher table

The screenshot shows the Microsoft SQL Server Management Studio interface. The Object Explorer on the left lists various databases and objects. The main pane contains a T-SQL script for inserting records into the Teacher table:

```
-- insert records into teacher table
insert into Teacher(teacherID,firstname,lastname,email)
values(401,'Dhana','Laxmi','dhana@gmail.com'),
(402,'Raja','Shekar','raja@gmail.com'),
(403,'Swapna','Reddy','swapna@gmail.com'),
(404,'Latha','Sura','latha@gmail.com'),
(405,'Rekha','Goud','rekha@gmail.com'),
(406,'Renuka','Goud','renuka@gmail.com'),
(407,'Sunitha','Shetty','sunitha@gmail.com'),
(408,'Anjali','Reddy','anjali@gmail.com'),
(409,'Anu','Laxmi','anu@gmail.com'),
(410,'Manjula','Shetty','manjulaa@gmail.com');
```

The status bar at the bottom indicates "Query executed successfully." and "Completion time: 2024-01-25T16:42:12.4027207+05:30".

Retrieving records from teacher table

The screenshot shows the Microsoft SQL Server Management Studio interface. The Object Explorer on the left lists various databases and objects. The main pane contains a T-SQL script for selecting all records from the Teacher table:

```
values(401,'Dhana','Laxmi','dhana@gmail.com'),
(402,'Raja','Shekar','raja@gmail.com'),
(403,'Swapna','Reddy','swapna@gmail.com'),
(404,'Latha','Sura','latha@gmail.com'),
(405,'Rekha','Goud','rekha@gmail.com'),
(406,'Renuka','Goud','renuka@gmail.com'),
(407,'Sunitha','Shetty','sunitha@gmail.com'),
(408,'Anjali','Reddy','anjali@gmail.com'),
(409,'Anu','Laxmi','anu@gmail.com'),
(410,'Manjula','Shetty','manjulaa@gmail.com');

select * from Teacher;
```

The results pane displays the retrieved data:

teacherID	firstname	lastname	email	
1	401	Dhana	Laxmi	dhana@gmail.com
2	402	Raja	Shekar	raja@gmail.com
3	403	Swapna	Reddy	swapna@gmail.com
4	404	Latha	Sura	latha@gmail.com
5	405	Rekha	Goud	rekha@gmail.com
6	406	Renuka	Goud	renuka@gmail.com
7	407	Sunitha	Shetty	sunitha@gmail.com
8	408	Anjali	Reddy	anjali@gmail.com
9	409	Anu	Laxmi	anu@gmail.com
10	410	Manjula	Shetty	manjulaa@gmail.com

The status bar at the bottom indicates "Query executed successfully." and "Completion time: 2024-01-25T16:42:12.4027207+05:30".

8) Insert records into Course table

```
codingchallenge.sql - DESKTOP-HPGQI4S\SQLEXPRESS.codingchallenge (DESKTOP-HPGQI4S\DELL (75)) - Microsoft SQL Server Management Studio
File Edit View Query Project Tools Window Help
Execute
select * from Teacher;

-- insert records into course table
insert into Course(courseID,CourseName,credits,teacherID)
values(201,'Java',3,402),
(202,'Python',4,403),
(203,'C',2,405),
(204,'C++',5,406),
(205,'R',1,401),
(206,'DBMS',0,402),
(207,'Operating Systems',3,403),
(208,'Design Analysis',1,407),
(209,'AI',2,409),
(210,'Machine Learning',4,410);

(10 rows affected)

Completion time: 2024-01-25 16:45:05.5429991+05:30
```

Query executed successfully.

Retrieving records from Course table

```
codingchallenge.sql - DESKTOP-HPGQI4S\SQLEXPRESS.codingchallenge (DESKTOP-HPGQI4S\DELL (75)) - Microsoft SQL Server Management Studio
File Edit View Query Project Tools Window Help
Execute
insert into Course(courseID,CourseName,credits,teacherID)
values(201,'Java',3,402),
(202,'Python',4,403),
(203,'C',2,405),
(204,'C++',5,406),
(205,'R',1,401),
(206,'DBMS',0,402),
(207,'Operating Systems',3,403),
(208,'Design Analysis',1,407),
(209,'AI',2,409),
(210,'Machine Learning',4,410);

-- retrieving records from course table
select * from Course;
```

courseID	CourseName	credits	teacherID
1	Java	3	402
2	Python	4	403
3	C	2	405
4	C++	5	406
5	R	1	401
6	DBMS	0	402
7	Operating Systems	3	403
8	Design Analysis	1	407
9	AI	2	409
10	Machine Learning	4	410

Query executed successfully.

9) Inserting records into Enrollment table

codingchallenge.sql - DESKTOP-HPGQI4S\SQLEXPRESS.codingchallenge (DESKTOP-HPGQI4S\DELL (75)) - Microsoft SQL Server Management Studio

-- retrieving records from course table
select * from Course;

-- inserting records into Enrollment table
insert into Enrollment(enrollmentID,studentID,CourseID,EnrollmentDate)
values (301,101,201,'2022-09-12'),
(302,104,203,'2021-11-21'),
(303,102,204,'2020-10-17'),
(304,108,206,'2021-09-13'),
(305,104,208,'2022-08-11'),
(306,109,209,'2023-07-12'),
(307,110,201,'2022-05-28'),
(308,102,202,'2021-06-26'),
(309,107,203,'2022-03-11'),
(310,106,205,'2022-02-19');

(10 rows affected)

Completion time: 2024-01-25 16:48:48.1314086+05:30

Query executed successfully.

DESKTOP-HPGQI4S\SQLEXPRESS - DESKTOP-HPGQI4S\DELL (75) codingchallenge 00:00:00 0 rows

Retrieving records from Enrollment Table

codingchallenge.sql - DESKTOP-HPGQI4S\SQLEXPRESS.codingchallenge (DESKTOP-HPGQI4S\DELL (75)) - Microsoft SQL Server Management Studio

-- retrieving records from Enrollment table
select * from Enrollment;

Results

enrollmentID	studentID	CourseID	EnrollmentDate	
1	301	101	201	2022-08-11
2	302	104	203	2021-11-21
3	303	105	204	2020-10-17
4	304	108	206	2021-09-13
5	305	104	208	2022-08-11
6	306	109	209	2023-07-12
7	307	110	201	2022-05-28
8	308	102	202	2021-06-26
9	309	107	203	2022-03-11
10	310	106	205	2022-02-19

Query executed successfully.

DESKTOP-HPGQI4S\SQLEXPRESS - DESKTOP-HPGQI4S\DELL (75) codingchallenge 00:00:00 10 rows

10) Inserting records into payment table

```

codingchallenge.sql - DESKTOP-HPGQI4S\SQLEXPRESS.codingchallenge (DESKTOP-HPGQI4S\DELL (75)) - Microsoft SQL Server Management Studio
File Edit View Project Tools Window Help
codingchallenge - Execute ✓
select * from Enrollment;
-- inserting records into payment table
insert into Payment(PaymentID,StudentID,Amount,PaymentDate)
values(501,101,8900,'2022-09-15'),
(502,104,9000,'2021-11-21'),
(503,102,7500,'2020-10-17'),
(504,107,6200,'2022-08-11'),
(505,108,29000,'2022-08-11'),
(506,109,12000,'2023-07-12'),
(507,110,43222,'2022-05-28'),
(508,102,5600,'2021-06-26'),
(509,103,8900,'2022-03-11'),
(510,105,9050,'2022-02-19');

(10 rows affected)

Completion time: 2024-01-25T16:53:04.7026495+05:30

```

Query executed successfully.

Retrieve records from Payment table

```

codingchallenge.sql - DESKTOP-HPGQI4S\SQLEXPRESS.codingchallenge (DESKTOP-HPGQI4S\DELL (75)) - Microsoft SQL Server Management Studio
File Edit View Project Tools Window Help
codingchallenge - Execute ✓
values(501,101,8900,'2022-09-15'),
(502,104,9000,'2021-11-21'),
(503,102,7500,'2020-10-17'),
(504,107,6200,'2022-08-11'),
(505,108,29000,'2022-08-11'),
(506,109,12000,'2023-07-12'),
(507,110,43222,'2022-05-28'),
(508,102,5600,'2021-06-26'),
(509,103,8900,'2022-03-11'),
(510,105,9050,'2022-02-19');

select * from Payment;

Results

```

PaymentID	StudentID	Amount	PaymentDate
1 501	101	8900	2022-09-15
2 502	104	9000	2021-11-21
3 503	102	7500	2020-10-17
4 504	107	6200	2022-08-11
5 505	108	29000	2022-08-11
6 506	109	12000	2023-07-12
7 507	110	43222	2022-05-28
8 508	102	5600	2021-06-26
9 509	103	8900	2022-03-11
10 510	105	9050	2022-02-19

Query executed successfully.

A) Execute OVER and PARTITION BY Clause in SQL Queries

- The 'PARTITION BY' clause in SQL is a subclause of the 'OVER' clause.
- It is used to split a large table into smaller, more manageable partitions to perform calculations easily.
- Each partition is then processed for the function present in the 'OVER()' clause.
- Partition is done based on the column name by which we want to divide into groups.

Syntax:

```
Select col_name,aggregate_function(col_name)
over(partition by col_name)
from table_name;
```

Example:

partition by studentid and find average of amount in payment table

The screenshot shows the Microsoft SQL Server Management Studio interface. In the Object Explorer, a database named 'codingchallenge' is selected. In the center pane, a query is written in T-SQL:

```
(510,105,9050,'2022-02-19');

select * from Payment;

-- A) partition by studentid and find average of amount in payment table
select paymentid,studentid,amount,paymentdate,
avg(amount) OVER(partition by studentid) as 'Average Amount of student by id'
from Payment;
```

The results grid displays the following data:

	paymentid	studentid	amount	paymentdate	Average Amount of student by id
1	501	101	8900	2022-06-15	8900
2	508	102	5000	2022-06-15	5000
3	503	103	7850	2020-10-17	6850
4	508	103	8900	2020-03-11	8900
5	502	104	9000	2021-11-21	9000
6	510	105	9050	2022-02-19	9050
7	504	107	6200	2022-08-11	6200
8	505	108	29000	2022-08-11	29000
9	506	109	12000	2022-07-12	12000
10	507	110	43222	2022-06-28	43222

At the bottom of the results grid, it says "Query executed successfully."

This query partitions by studentid to find the average of amounts in payments. It gives paymentid, studentid, amount, paymentdate and the average amount obtained by partitioning or dividing into groups by studentid.

In this query, partitioning is done by studentid.

B) Creating Sub Totals

- Sub total is the sum of similar sets of data but it does not indicate the final total.
- Calculating a subtotal in SQL query can be a bit complicated than the common aggregate queries
- Sub totals can be calculated using a roll up group by extension.

- The **ROLLUP** extension allows us to generate hierarchical subtotal rows according to its input columns and it also adds a grand total row to the result set.

Example:

The screenshot shows the Microsoft SQL Server Management Studio interface. In the Object Explorer, a database named 'codingchallenge' is selected. In the center pane, a query is written:

```

-- B) Calculating sub totals
-- Subtotals are calculated using ROLL UP extension
-- Example
SELECT CourseID, StudentID, COUNT(*) AS EnrollmentCount
FROM Enrollment
GROUP BY ROLLUP (CourseID, StudentID);

```

The Results pane displays the output of the query:

CourseID	StudentID	EnrollmentCount
1	201	101
2	201	101
3	201	NULL
4	202	102
5	202	NULL
6	203	104
7	203	107
8	203	NULL
9	204	102
10	204	NULL
11	205	106
12	205	NULL
13	206	108
14	206	NULL
15	206	104
16	206	NULL
17	209	109
18	209	NULL
19	NULL	10

Below the results, a message states "Query executed successfully."

In this query to calculate sub total group by rollup is used where group by is done using CourseID and StudentID which gives the subtotal and total count of students enrolled in courses.All the subtotals are summed together to give grand total.

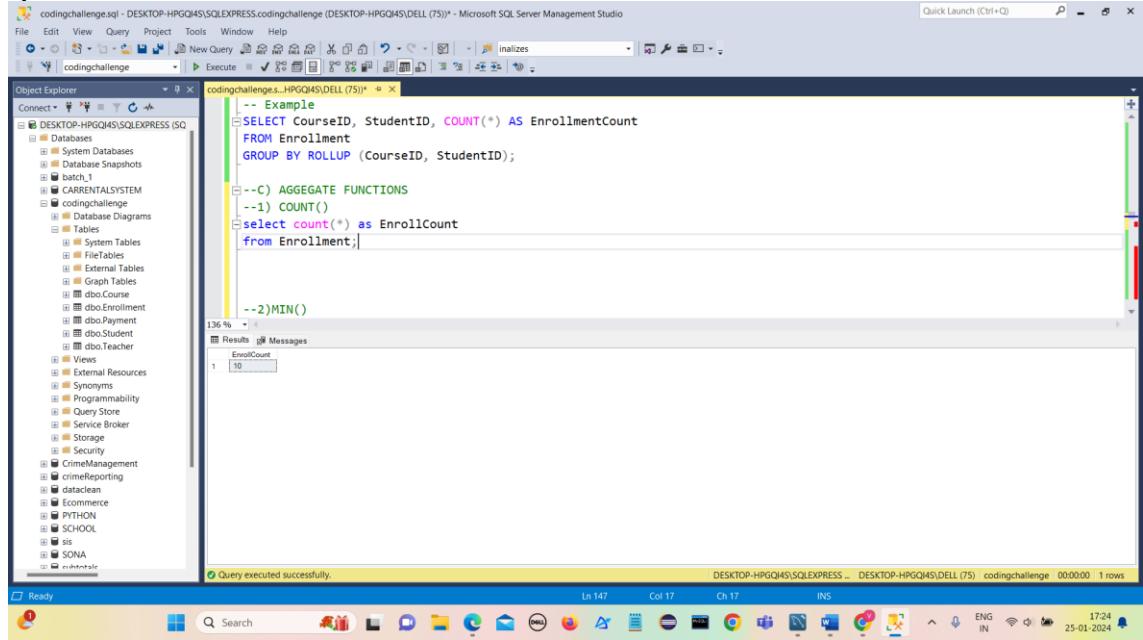
C) Total Aggregations in SQL

An SQL aggregate function calculates operations on a column of values and returns a single value.

Aggregate functions include:

- 1) **COUNT()**
- 2) **SUM()**
- 3) **MIN()**
- 4) **MAX()**
- 5) **AVG()**

- 1) COUNT()-The COUNT() function returns the number of rows that matches a specified criterion.**



```
-- Example
SELECT CourseID, StudentID, COUNT(*) AS EnrollmentCount
FROM Enrollment
GROUP BY ROLLUP (CourseID, StudentID);

--C) AGGREGATE FUNCTIONS
--1) COUNT()
select count(*) as EnrollCount
from Enrollment;

--2) MIN()

```

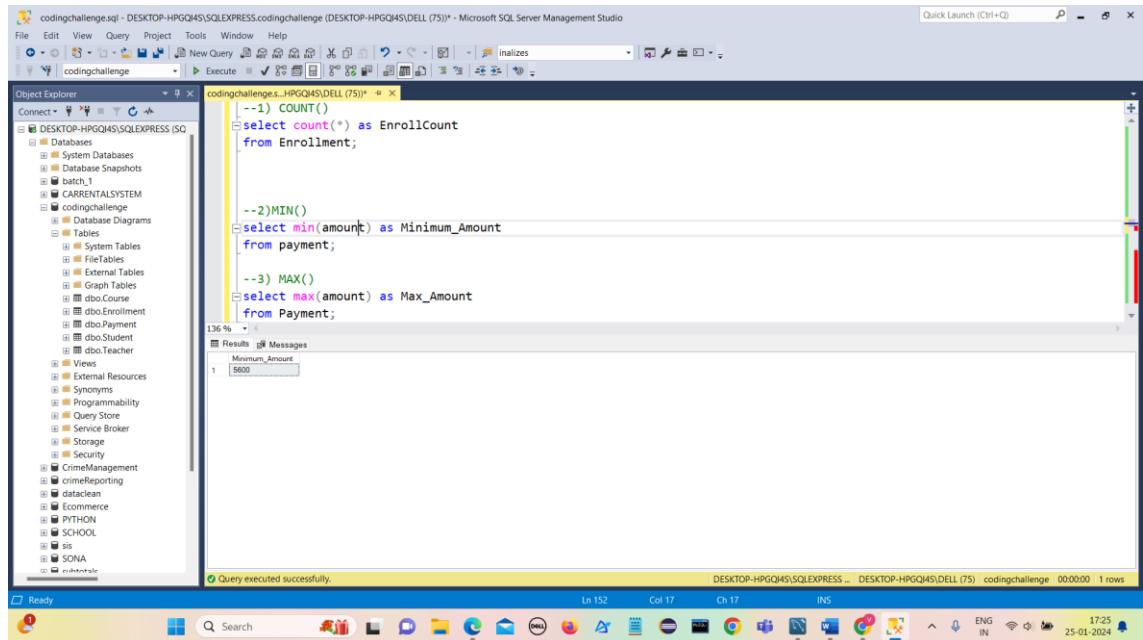
Results Messages

EnrollCount
10

Query executed successfully.

This query gives the total records in enrolment table

- 2) MIN()-The MIN() function returns the smallest value of the selected column.**



```
--1) COUNT()
select count(*) as EnrollCount
from Enrollment;

--2) MIN()
select min(amount) as Minimum_Amount
from payment;

--3) MAX()
select max(amount) as Max_Amount
from Payment;
```

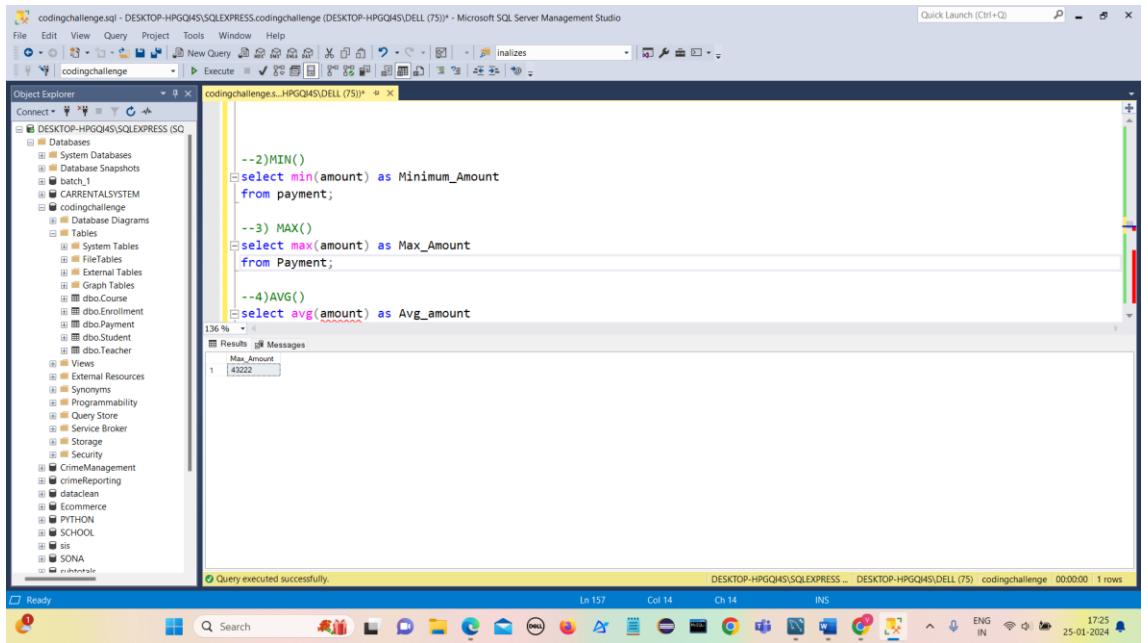
Results Messages

Minimum_Amount
5600

Query executed successfully.

This query gives the minimum of salary of payments

3) MAX()-The MAX() function returns the largest value of the selected column.



The screenshot shows the Microsoft SQL Server Management Studio interface. On the left, the Object Explorer pane displays the database structure, including the 'codingchallenge' database and its tables like 'batch_1', 'CURRENTALSYSTEM', and 'Payment'. In the center, a query window titled 'codingchallenge.s...HPGQ4S(DELL (75))' contains the following T-SQL code:

```
--2)MIN()
select min(amount) as Minimum_Amount
from payment;

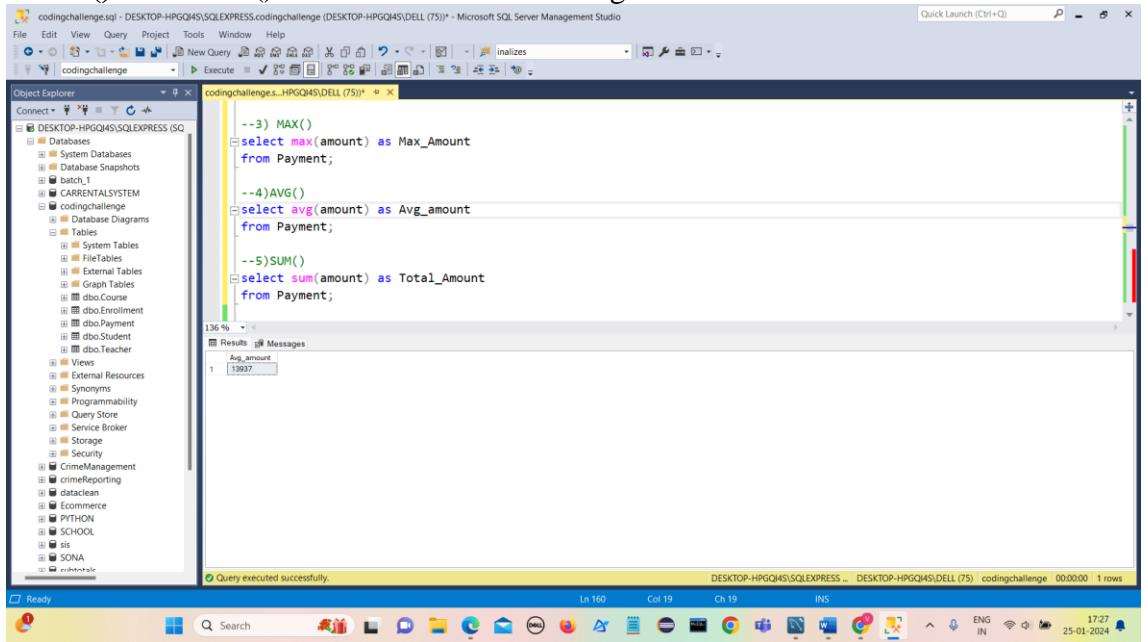
--3) MAX()
select max(amount) as Max_Amount
from Payment;

--4)AVG()
select avg(amount) as Avg_amount
```

The results pane shows a single row with the value '43222' under the 'Max_Amount' column. At the bottom of the query window, it says 'Query executed successfully.'

This query gives the maximum of salary of payments

4) AVG()-The AVG() function returns the average value of a numeric column



The screenshot shows the Microsoft SQL Server Management Studio interface. On the left, the Object Explorer pane displays the database structure, including the 'codingchallenge' database and its tables like 'batch_1', 'CURRENTALSYSTEM', and 'Payment'. In the center, a query window titled 'codingchallenge.s...HPGQ4S(DELL (75))' contains the following T-SQL code:

```
--3) MAX()
select max(amount) as Max_Amount
from Payment;

--4)AVG()
select avg(amount) as Avg_amount
from Payment;

--5)SUM()
select sum(amount) as Total_Amount
from Payment;
```

The results pane shows a single row with the value '13937' under the 'Avg_amount' column. At the bottom of the query window, it says 'Query executed successfully.'

This query gives the average of salary of payments

5) SUM()-The SUM() function returns the total sum of a numeric column.

The screenshot shows the Microsoft SQL Server Management Studio interface. In the Object Explorer on the left, there is a tree view of database objects under the 'codingchallenge' database. In the center, a query window titled 'codingchallenge..HPGQ4S(DELL (75))' displays the following T-SQL code:

```
from Payment;
--4)AVG()
select avg(amount) as Avg_amount
from Payment;

--5)SUM()
select sum(amount) as Total_Amount
from Payment;]
```

The results grid on the right shows a single row with the value '139372' under the column 'Total_Amount'. Below the results grid, a message says 'Query executed successfully.' At the bottom of the screen, the taskbar shows various application icons and the system clock indicating '17:27 25-01-2024'.

This query gives the sum of salary of payments