EXERCISE-6

Single Row Functions

Objective

After the completion of this exercise, the students will be able to do the following:

- Describe various types of functions available in SQL.
- Use character, number and date functions in SELECT statement.
- Describe the use of conversion functions.

Single row functions:

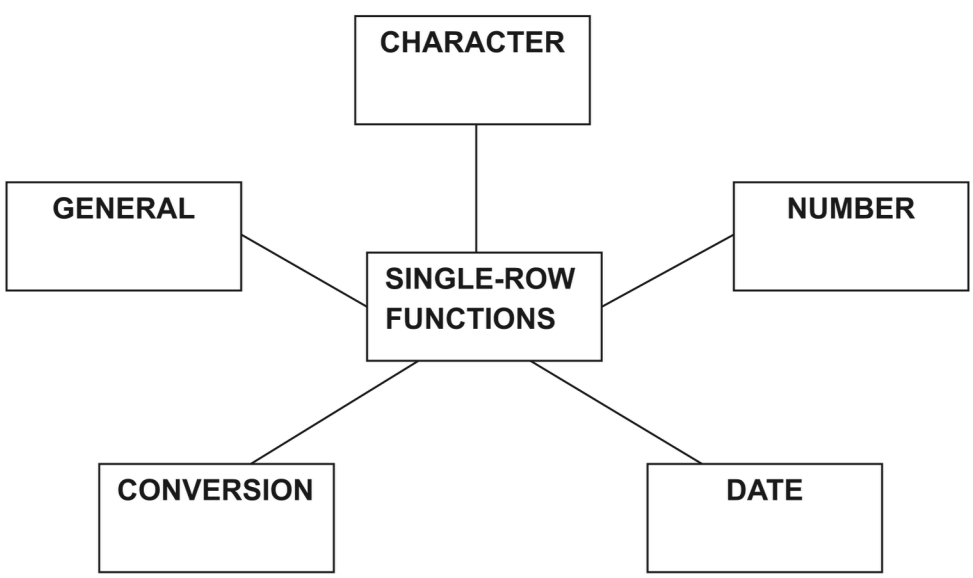
Manipulate data items.
Accept arguments and return one value.
Act on each row returned.
Return one result per row.
May modify the data type.
Can be nested.
Accept arguments which can be a column or an expression

Syntax

Function_name(arg1,…argn)

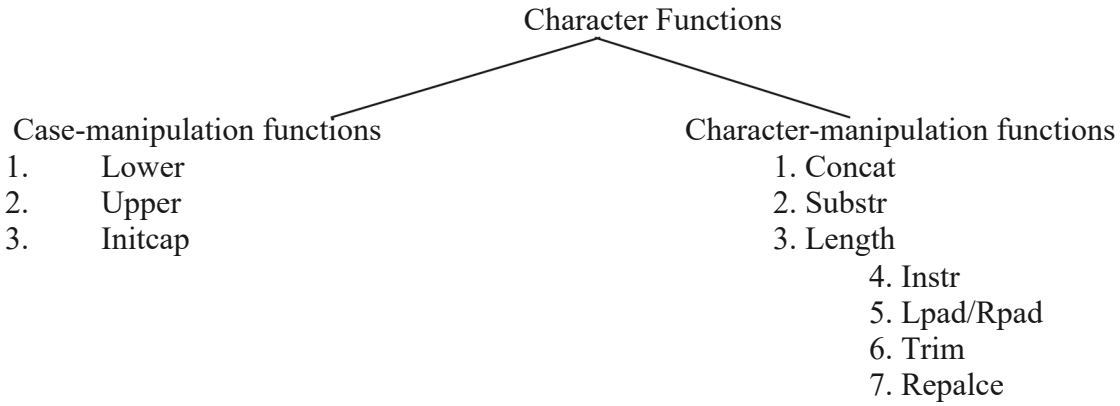
An argument can be one of the following

- ✓ User-supplied constant
- ✓ Variable value
- ✓ Column name
- ✓ Expression



- Character Functions: Accept character input and can return both character and number values.
- Number functions: Accept numeric input and return numeric values.
- Date Functions: Operate on values of the DATE data type.
- Conversion Functions: Convert a value from one type to another.

Character Functions



Function	Purpose
lower(column/expr) upper(column/expr)	Converts alpha character values to lowercase
initcap(column/expr)	Converts alpha character values to uppercase
concat(column1/expr1, column2/expr2)	Converts alpha character values the to uppercase for the first letter of each word, all other letters in lowercase
substr(column/expr,m,n)	Concatenates the first character to the second character
length(column/expr)	Returns specified characters from character value starting at character position m, n characters long
instr(column/expr,'string',m,n)	Returns the number of characters in the expression
lpad(column/expr, n,'string')	Returns the numeric position of a named string
rpad(column/expr,'string',m,n)	Pads the character value right-justified to a total width of n character positions
trim(leading/trailing/both, trim_character FROM trim_source)	Pads the character value left-justified to a total width of n character positions
replace(text, search_string, replacement_string)	Enables you to trim heading or string. trailing or both from a character

Example:

lower('SQL Course') sql course

upper('SQL Course') SQL COURSE

initcap('SQL Course') Sql Course

```
SELECT 'The job id for'|| upper(last_name||'is'||lower(job_id) AS "EMPLOYEE DETAILS"
FROM employees;
SELECT employee_id, last_name, department_id
FROM employees
WHERE LOWER(last_name)='higgins';
```

Function	Result
CONCAT('hello', 'world')	helloworld
Substr('helloworld',1,5)	Hello 10 6
Length('helloworld')	*****24000
Instr('helloworld','w')	24000*****
Lpad(salary,10,'*')	elloworld
Rpad(salary,10,'*') Trim('h'	
FROM 'helloworld')	

Command	Query	Output
initcap(char);	*select initcap("hello") from dual;*	Hello
lower (char); upper (char);	*select lower ('HELLO') from dual;* *select upper ('hello') from dual;*	Hello HELLO
ltrim (char,[set]);	*select ltrim ('cseit', 'cse') from dual;*	IT
rtrim (char,[set]);	*select rtrim ('cseit', 'it') from dual;*	CSE
replace (char,search string, replace string);	*select replace ('jack and jue', 'j', 'bl') from dual;*	black and blue
substr (char,m,n);	*select substr ('information', 3, 4) from dual;*	form

Example:

SELECT employee_id, CONCAT (first_name,last_name) NAME , job_id,LENGTH(last_name), INSTR(last_name,'a') "contains'a'?"

FROMemployeesWHERESUBSTR(job_id,4)='ERP';

NUMBER FUNCTIONS

Function	Purpose
round(column/expr, n)	Rounds the value to specified decimal
trunc(column/expr,n)	Truncates value to specified decimal
mod(m,n)	Returns remainder of division

Example

Function	Result
round(45.926,2)	45.93
trunc(45.926,2)	45.92
mod(1600,300)	100

SELECT ROUND(45.923,2), ROUND(45.923,0), ROUND(45.923,-1) FROM dual;

NOTE: Dual is a dummy table you can use to view results from functions and calculations.

SELECT TRUNC(45.923,2), TRUNC(45.923), TRUNC(45.923,-2) FROM dual;

SELECT last_name,salary,MOD(salary,5000) FROM employees WHERE job_id='sa_rep';

Working with Dates

minutes, and seconds. • The default date display format is DD-MON-RR. – Enables you to store 21st-century dates in the 20th century by specifying only the last

two digits of the year
– Enables you to store 20th-century dates in the 21st century in the same way

Example

SELECT last_name, hire_date FROM employees WHERE hire_date < '01-FEB-88;

Working with Dates

SYSDATE is a function that returns:
• Date
• Time

Example

Display the current date using the DUAL table.

SELECT SYSDATE FROM DUAL;

Arithmetic with Dates

• Add or subtract a number to or from a date for a resultant date value.
• Subtract two dates to find the number of days between those dates.
• Add hours to a date by dividing the number of hours by 24.

Arithmetic with Dates

Because the database stores dates as numbers, you can perform calculations using arithmetic Operators such as addition and subtraction. You can add and subtract number constants as well as dates.
You can perform the following operations:

Operation	Result	Description
date + number	Date	Adds a number of days to a date
date – number	Date	Subtracts a number of days from a date
date – date	Number of days	Subtracts one date from another
date + number/24	Date	Adds a number of hours to a date

Example

SELECT last_name, (SYSDATE-hire_date)/7 AS WEEKS
FROM employees WHERE department_id = 90;

Date Functions

Function	Result
MONTHS_BETWEEN	Number of months between two dates
ADD_MONTHS	Add calendar months to date
NEXT_DAY	Next day of the date specified
LAST_DAY	Last day of the month
ROUND	Round date
TRUNC	Truncate date

Date Functions

except MONTHS_BETWEEN, which returns a numeric value.

• MONTHS_BETWEEN(date1, date2):::: Finds the number of months between date1 and date2. The result can be positive or negative. If date1 is later than date2, the result is positive; if date1 is earlier than date2, the result is negative. The noninteger part of the result represents a portion of the month.
• ADD_MONTHS(date, n):::: Adds n number of calendar months to date. The value of n must be an integer and can be negative.
• NEXT_DAY(date, 'char'):::: Finds the date of the next specified day of the week ('char') following date. The value of char may be a number representing a day or a character string.
• LAST_DAY(date):::: Finds the date of the last day of the month that contains date
• ROUND(date[,'fmt']):::: Returns date rounded to the unit that is specified by the format model fmt. If the format model fmt is omitted, date is rounded to the nearest day.

• TRUNC(date[, 'fmt']):::: Returns date with the time portion of the day truncated to the unit that is specified by the format model fmt. If the format model fmt is omitted, date is truncated to the nearest day.

Using Date Functions

Function	Result
MONTHS_BETWEEN ('01-SEP-95','11-JAN-94')	19.6774194
ADD_MONTHS ('11-JAN-94',6)	'11-JUL-94'
NEXT_DAY ('01-SEP-95','FRIDAY')	'08-SEP-95'
LAST_DAY ('01-FEB-95')	'28-FEB-95'

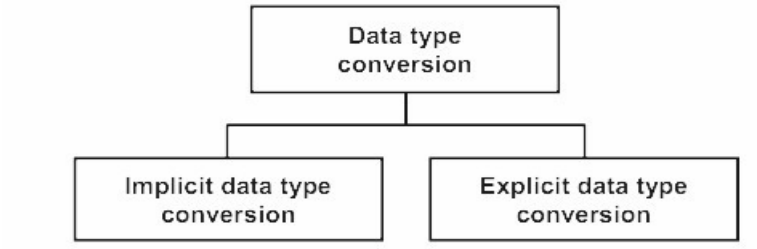
Example

Display the employee number, hire date, number of months employed, sixmonth review date, first Friday after hire date, and last day of the hire month for all employees who have been employed for fewer than 70 months.

```
SELECT employee_id, hire_date,MONTHS_BETWEEN (SYSDATE, hire_date)
TENURE,ADD_MONTHS (hire_date, 6) REVIEW,NEXT_DAY (hire_date, 'FRIDAY'),
LAST_DAY(hire_date)
FROM employees
WHERE MONTHS_BETWEEN (SYSDATE, hire_date) < 70;
```

Conversion Functions

This covers the following topics:
• Writing a query that displays the current date
• Creating queries that require the use of numeric, character, and date functions
• Performing calculations of years and months of service for an employee



Implicit Data Type Conversion

For assignments, the Oracle server can automatically convert the following:

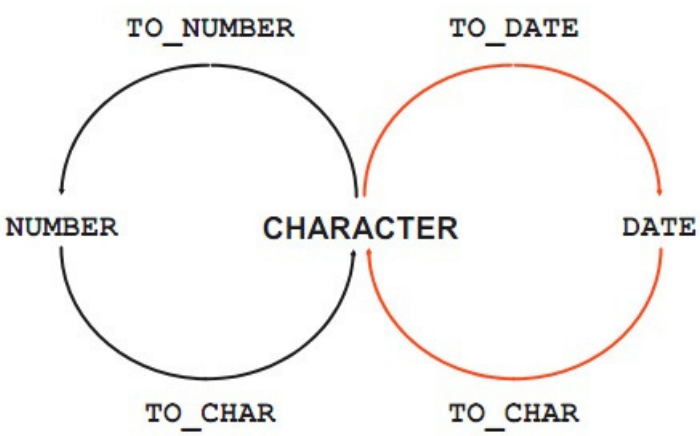
From	To
VARCHAR2 or CHAR	NUMBER
VARCHAR2 or CHAR	DATE
NUMBER	VARCHAR2
DATE	VARCHAR2

For example, the expression hire_date > '01-JAN-90' results in the implicit conversion from the string '01-JAN-90' to a date.

For expression evaluation, the Oracle Server can automatically convert the following:

From	To
VARCHAR2 or CHAR	NUMBER
VARCHAR2 or CHAR	DATE

Explicit Data Type Conversion



SQL provides three functions to convert a value from one data type to another:

Example:

Using the TO_CHAR Function with Dates

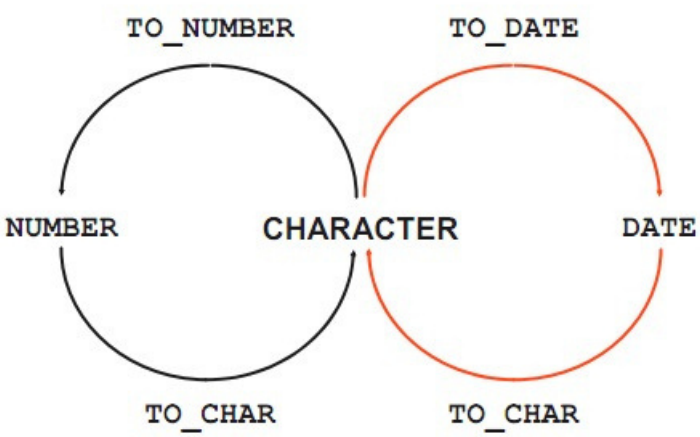
TO_CHAR(date,'format_model')

The format model:

- Mustbeenclosed bysinglequotationmarks
- Is case-sensitive

• Has an fm element to remove padded blanks or suppress leading zeros • Is separated from the date value by a comma SELECT employee_id, TO_CHAR(hire_date, 'MM/YY') Month_Hired FROM employees WHERE last_name = 'Higgins';

Elements of the Date Format Model



Element	Description
SCC or CC	Century; server prefixes B.C. date with -
Years in dates YYYY or SYYYY	Year; server prefixes B.C. date with -
YYY or YY or Y	Last three, two, or one digits of year
Y,YYY	Year with comma in this position
IYYY, IYY, IY, I	Four-, three-, two-, or one-digit year based on the ISO standard
SYEAR or YEAR	Year spelled out; server prefixes B.C. date with -
BC or AD	Indicates B.C. or A.D. year
B.C. or A.D.	Indicates B.C. or A.D. year using periods
Q	Quarter of year
MM	Month: two-digit value
MONTH	Name of month padded with blanks to length of nine characters
MON	Name of month, three-letter abbreviation
RM	Roman numeral month
WW or W	Week of year or month
DDD or DD or D	Day of year, month, or week
DAY	Name of day padded with blanks to a length of nine characters
DY	Name of day; three-letter abbreviation
J	Julian day; the number of days since December 31, 4713 B.C.

Date Format Elements: Time Formats

Use the formats that are listed in the following tables to display time information and literals and to change numerals to spelled numbers.

Element	Description
AM or PM	Meridian indicator
A.M. or P.M.	Meridian indicator with periods
HH or HH12 or HH24	Hour of day, or hour (1–12), or hour (0–23)
MI	Minute (0–59)
SS	Second (0–59)
SSSSS	Seconds past midnight (0–86399)

Other Formats

Element	Description
/ . ,	Punctuation is reproduced in the result.
"of the"	Quoted string is reproduced in the result.

Specifying Suffixes to Influence Number Display

Element	Description
TH	Ordinal number (for example, DDTH for 4TH)
SP	Spelled-out number (for example, DDSP for FOUR)
SPTH or THSP	Spelled-out ordinal numbers (for example, DDSPTH for FOURTH)

Example

```
SELECTlast_name,
TO_CHAR(hire_date, 'fmDD Month YYYY') AS HIREDATE
```

Modify example to display the dates in a format that appears as “Seventeenth of June 1987 12:00:00 AM.” SELECT last_name, TO_CHAR (hire_date, 'fmDdspth "of" Month YYYY fmHH:MI:SS AM') HIREDATE FROM employees;

Using the TO_CHAR Function with Numbers

TO_CHAR(number, 'format_model') These are some of the format elements that you can use with the TO_CHAR function to display a number value as a character:

Element	Result
9	Represents a number
0	Forces a zero to be displayed
$	Places a floating dollar sign
L	Uses the floating local currency symbol
.	Prints a decimal point
,	Prints a comma as thousands indicator

Number Format Elements

If you are converting a number to the character data type, you can use the following format elements:

Element	Description	Example	Result
9	Numeric position (number of 9s determine display width)	999999	1234
0	Display leading zeros	099999	001234
$	Floating dollar sign	$999999	$1234
L	Floating local currency symbol	L999999	FF1234
D	Returns in the specified position the decimal character. The default is a period (.).	99D99	99.99
.	Decimal point in position specified	999999.99	1234.00
G	Returns the group separator in the specified position. You can specify multiple group separators in a number format model.	9,999	9G999
,	Comma in position specified	999,999	1,234
MI	Minus signs to right (negative values)	999999MI	1234-
PR	Parenthesize negative numbers	999999PR	<1234>
EEEE	Scientific notation (format must specify four Es)	99.999EEEE	1.234E+03
U	Returns in the specified position the "Euro" (or other) dual currency	U9999	€1234
V	Multiply by 10 *n* times (*n* = number of 9s after V)	9999V99	123400
S	Returns the negative or positive value	S9999	-1234 or +1234
B	Display zero values as blank, not 0	B9999.99	1234.00

SELECT TO_CHAR(salary, '$99,999.00') SALARY

FROM employees WHERE last_name ='Ernst';

Using the TO_NUMBER and TO_DATE Functions

• Convert a character string to a number format using the TO_NUMBER function: TO_NUMBER(char[, 'format_model'] • Convert a character string to a date format using the TO_DATE function: TO_DATE(char[, 'format_model'] • These functions have an fx modifier. This modifier specifies the exact matching for the character

argument and date format model of a TO_DATE function.

The fx modifier specifies exact matching for the character argument and date format model of a TO_DATE function:

• Punctuation and quoted text in the character argument must exactly match (except for case) the corresponding parts of the format model.

• The character argument cannot have extra blanks. Without fx, Oracle ignores extra blanks.

• Numeric data in the character argument must have the same number of digits as the corresponding element in the format model. Without fx, numbers in the character argument can omit leading zeros.

```
SELECT last_name, hire_date
FROM employees
WHERE hire_date = TO_DATE('May 24, 1999', 'fxMonth DD, YYYY');
```

Find the Solution for the following:

1. Write a query to display the current date. Label the column Date.

```
SELECT SYSDATE "DATE"
FROM DUAL;
```

Download ▾ Execution time: 0.002 seconds

	DATE
1	9/17/2025, 5:02:10

2. The HR department needs a report to display the employee number, last name, salary, and increased by 15.5% (expressed as a whole number) for each employee. Label the column New Salary.

```
select id,last_name,salary,
round(salary*1.155)"New Salary"
FROM MY_EMPLOYEE;
```

Download ▾ Execution time: 0.108 seconds

	ID	LAST_NAME	SALARY	NEW SALARY
1	101	Doe	50000	57750
2	102	Sharma	61450.25	70975
3	103	Zhang	45400.5	52438
4	104	Ortiz	71000	82005
5	105	Khan	52350.75	60465

3. Modify yourquerylab_03_02.sqltoadd a column that subtracts the old salary from the new salary. Label the column Increase.

```
select id,last_name,salary,
round(salary*1.155)"New Salary",
round(salary*1.155)-salary"Increase"
FROM MY_EMPLOYEE;
```

Download Execution time: 0.006 seconds

	ID	LAST_NAME	SALARY	NEW SALARY	INCREASE
1	101	Doe	50000	57750	7750
2	102	Sharma	61450.25	70975	9524.75
3	103	Zhang	45400.5	52438	7037.5
4	104	Ortiz	71000	82005	11005
5	105	Khan	52350.75	60465	8114.25

4. Write a query that displays the last name (with the first letter uppercase and all other letters lowercase) and the length of the last name for all employees whose name starts with the letters J, A, or M. Give each column an appropriate label. Sort the results by the employees' last names.

```
select initcap(last_name)"LAST NAME",
length(last_name)"LENGTH"
FROM MY_EMPLOYEE where upper(substr(last_name,1,1))
in ('J','A','M')
Order by last_name;
```

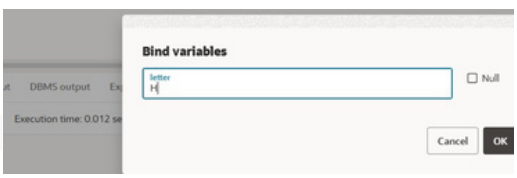
Download Execution time: 0.01 seconds

LAST NAME	LENGTH

No items to display.

5. Rewrite the query so that the user is prompted to enter a letter that starts the last name. For example, if the user enters H when prompted for a letter, then the output should show all employees whose last name starts with the letter H.

```
select first_name,last_name
from MY_EMPLOYEE
where last_name like :letter || '%';
```



6. The HR department wants to find the length of employment for each employee. For each employee, display the last name and calculate the number of months between today and the date on which the employee was hired. Label the column MONTHS_WORKED. Order your results by the number of months employed. Round the number of months up to the closest whole number.

```
select last_name,round(months_between(SYSDATE,hire_date))
As "Months Worked"
from MY_EMPLOYEE
Order by round(months_between(SYSDATE,hire_date));
```

	LAST_NAME	MONTHS WORKED
1	Khan	50
2	Ortiz	56
3	Sharma	65
4	Doe	70
5	Zhang	88

Note: Your results will differ.

7. Create a report that produces the following for each employee:
<employee last name> earns <salary> monthly but wants <3 times salary>. Label the column Dream Salaries.

```
select last_name || ' earns ' || salary ||
' monthly but wants ' || (salary*3)
AS "Dream Salaries"
FROM MY_EMPLOYEE;
```

Download Execution time: 0.003 seconds

	DREAM SALARIES
1	Doe earns 50000 monthly but wants 150000
2	Sharma earns 61450.25 monthly but wants 184350.75
3	Zhang earns 45400.5 monthly but wants 136201.5
4	Ortiz earns 71000 monthly but wants 213000
5	Khan earns 52350.75 monthly but wants 157052.25

8. Create a query to display the last name and salary for all employees. Format the salary to be 15 characters long, left-padded with the $ symbol. Label the column SALARY.

```
select last_name,
Lpad(salary,15,'$')"Salary"
FROM MY_EMPLOYEE;
```

Download Execution time: 0.005 seconds

	LAST_NAME	SALARY
1	Doe	$$$$$$$$$50000
2	Sharma	$$$$$$$61450.25
3	Zhang	$$$$$$$$45400.5
4	Ortiz	$$$$$$$$$71000
5	Khan	$$$$$$$52350.75

9. Display each employee's last name, hire date, and salary review date, which is the first Monday after six months of service. Label the column REVIEW. Format the dates to appear in the format similar to "Monday, the Thirty-First of July, 2000."

```
select last_name, hire_date,
NEXT_DAY(ADD_MONTHS(hire_date,6),'Monday')
AS "REVIEW"
FROM MY_EMPLOYEE;
```

Download Execution time: 0.102 seconds

	LAST_NAME	HIRE_DATE	REVIEW
1	Doe	11/15/2019, 12:00:00 AM	5/18/2020, 12:00:00 AM
2	Sharma	4/22/2020, 12:00:00 AM	10/26/2020, 12:00:00 AM
3	Zhang	5/9/2018, 12:00:00 AM	11/12/2018, 12:00:00 AM
4	Ortiz	1/19/2021, 12:00:00 AM	7/26/2021, 12:00:00 AM
5	Khan	7/10/2021, 12:00:00 AM	1/17/2022, 12:00:00 AM

10. Display the last name, hire date, and day of the week on which the employee started. Label the column DAY. Order the results by the day of the week, starting with Monday.

```
select last_name, hire_date,
to_char(hire_date,'Day')AS "DAY"
FROM MY_EMPLOYEE
Order by to_char(hire_date,'D');
```

Download Execution time: 0.005 seconds

	LAST_NAME	HIRE_DATE	DAY
1	Ortiz	1/19/2021, 12:00:00 AM	Tuesday
2	Sharma	4/22/2020, 12:00:00 AM	Wednesday
3	Zhang	5/9/2018, 12:00:00 AM	Wednesday
4	Doe	11/15/2019, 12:00:00 AM	Friday
5	Khan	7/10/2021, 12:00:00 AM	Saturday

Evaluation Procedure	Marks awarded
Query(5)	
Execution (5)	
Viva(5)	
Total (15)	
Faculty Signature	