MORE ON DATABASES AND SQL

Learning Objectives

After studying this lesson the students will be able to:

- Define the terms:
  - (i) Group (Aggregate) functions, Constraints
  - (ii) Cartesian Product, Join, Referential Integrity, Foreign Key.
- Write queries using aggregate functions and GROUP BY clause.
- Access data from multiple tables
- Create tables with PRIMARY KEY and NOT NULL constraints
- Add a constraint to a table, remove a constraint from a table, modify a column of a table using ALTER TABLE command.
- Delete a table using DROP TABLE.

In the previous class, you have learnt some database concepts and SQL commands. You have also learnt how to create databases and tables within databases and how to access data from a table using various clauses of SELECT command. In this chapter you shall learn some more clauses and functions in SQL to access data from a table and how to access data from multiple tables of a database.

Puzzle [8]

It was Iftar party in Lucknow that Mr. David met Mr. Naqvi. They became friends and exchanged their phone numbers. After a few days, Mr. David rang up and invited Mr. Naqvi for New Year party at his house and gave him his house number as follows:

"I live in a long street. Numbered on the side of my house are the houses one, two, three and so on. All the numbers on one side of my house add up to exactly the same as all the

numbers on the other side of my house. I know there are more than thirty houses on that side of the street, but not so many as 50."

With this information, Mr. Naqvi was able to find Mr. David's house number. Can you also find?

Such situations are faced by the developers of RDBMS software where they have to think of retrieval of data from multiple tables satisfying some specified conditions.

Let us now move ahead with SQL and more database concepts.

Ms. Shabana Akhtar is in-charge of computer department in a Shoe factory. She has created a database 'Shoes' with the following tables:

SHOES

(To store the information about various types of shoes made in the factory)

Field	Type	Null	Key	Default	Extra
code	char(4)	NO	PRI	NULL	
name	varchar(20)	YES		NULL	
type	varchar(10)	YES		NULL	
size	int(2)	YES		NULL	
cost	decimal(6,2)	YES		NULL	
margin	decimal(4,2)	YES		NULL	
Qty	int(4)	YES		NULL	

CUSTOMERS

(To store the data of customers)

Field	Type	Null	Key	Default	Extra
cust_Code	char(4)	NO	PRI	NULL	
name	varchar(30)	YES		NULL	
address	varchar(50)	YES		NULL	
phone	varchar(30)	YES		NULL	
category	char(1)	YES		NULL	

ORDERS

(To store the data of orders placed by customers)

Field	Type	Null	Key	Default	Extra
order_no	int(5)	NO	PRI	NULL	
cust_code	char(4)	YES		NULL	
Shoe_Code	char(4)	YES		NULL	
order_qty	int(4)	YES		NULL	
order_date	date	YES		NULL	
target_date	date	YES		NULL	

Sample data stored in these tables is given below:

SHOES

Code	Name	type	size	cost	margin	Qty
1001	School Canvas	School	6	132.50	2.00	1200
1002	School Canvas	School	7	135.50	2.00	800
1003	School Canvas	School	8	140.75	2.00	600
1011	School Leather	School	6	232.50	2.00	2200
1012	School Leather	School	7	270.00	2.00	1280
1013	School Leather	School	8	320.75	NULL	1100
1101	Galaxy	Office	7	640.00	3.00	200
1102	Galaxy	Office	8	712.00	3.00	500
1103	Galaxy	Office	9	720.00	3.00	400
1201	Tracker	Sports	6	700.00	NULL	280
1202	Tracker	Sports	7	745.25	3.50	NULL
1203	Tracker	Sports	8	800.50	3.50	600
1204	Tracker	Sports	9	843.00	NULL	860

CUSTOMERS

Cust_Code	name	address	Phone	Category
C001	Novelty Shoes	Raja Nagar, Bhopal	4543556, 97878989	A
C002	Aaram Footwear	31, Mangal Bazar, Agra	NULL	B
C003	Foot Comfort	New Market, Saharanpur	51917142, 76877888	B
C004	Pooja Shoes	Janak Puri, New Delhi	61345432, 98178989	A
C005	Dev Shoes	Mohan Nagar, Ghaziabad	NULL	C

ORDERS

order_no	cust_code	Shoe_Code	order_qty	order_date	target_date
1	C001	1001	200	2008-12-10	2008-12-15
2	C001	1002	200	2008-12-10	2008-12-15
3	C003	1011	150	2009-01-08	2009-01-13
4	C002	1012	250	2009-01-08	2009-01-13
5	C001	1011	400	2009-01-10	2009-01-15
6	C002	1101	300	2009-01-10	2009-01-15
7	C004	1201	200	2009-01-10	2009-01-15
8	C005	1102	350	2009-01-10	2009-01-15
9	C001	1103	225	2009-01-13	2009-01-18
10	C002	1203	200	2009-01-14	2009-01-19

Let us now see how this database helps Ms. Akhtar in generating various reports quickly.

Aggregate Functions

In class XI we studied about single row functions available in SQL. A single row function works on a single value. SQL also provides us multiple row functions. A multiple row function works on multiple values. These functions are called aggregate functions or group functions. These functions are:

S.No.	Function	Purpose
1	MAX()	Returns the MAXIMUM of the values under the specified column/expression.
2	MIN()	Returns the MINIMUM of the values under the specified column/expression.

3	AVG()	Returns the AVERAGE of the values under the specified column/expression.
4	SUM()	Returns the SUM of the values under the specified column/expression.
5	COUNT()	Returns the COUNT of the number of values under the specified column/expression.

MAX() :

MAX() function is used to find the highest value of any column or any expression based on a column. MAX() takes one argument which can be any column name or a valid expression involving a column name. e.g.,

Purpose	Statement	Output

To find the highest cost of any type of shoe in the factory.	`SELECT MAX(cost) FROM shoes;`	`+-----------+`
		`| MAX(cost) |`
		`+-----------+`
		`| 843.00 |`
		`+-----------+`

To find the highest cost of any shoe of type 'School'.	`SELECT MAX(cost) FROM shoes WHERE type = 'School';`	`+-----------+`
		`| MAX(cost) |`
		`+-----------+`
		`| 320.75 |`
		`+-----------+`

To find the highest selling price of any type of shoe.	`SELECT MAX(cost+cost*margin/100) FROM shoes;`	`+--------------------------+`
		`| MAX(cost+cost*margin/100) |`
		`+--------------------------+`
		`| 828.517500000 |`
		`+--------------------------+`

To find the highest selling price of any type of shoe rounded to 2 decimal places.	`SELECT ROUND(MAX(cost+cost*margin/100),2) AS "Max. SP" FROM shoes;`	`+---------+`
		`| Max. SP |`
		`+---------+`
		`| 733.36 |`
		`+---------+`

To find the highest selling price of any type of shoe rounded to 2 decimal places.	`SELECT ROUND(MAX(cost+cost*margin/100),2) AS "Max. SP" FROM shoes;`	`+---------+`
		`| Max. SP |`
		`+---------+`
		`| 733.36 |`
		`+---------+`

MIN() :

MIN() function is used to find the lowest value of any column or an expression based on a column. MIN() takes one argument which can be any column name or a valid expression involving a column name. e.g.,

Purpose	Statement	Output

To find the lowest cost of any type of shoe in the factory.	`SELECT MIN(cost) FROM shoes;`	`+-----------+`
		`| MIN(cost) |`
		`+-----------+`
		`| 843.00 |`
		`+-----------+`

To find the lowest cost of any shoe of type 'School'.	`SELECT MIN(cost) FROM shoes WHERE type = 'School';`	`+-----------+`
		`| MIN(cost) |`
		`+-----------+`
		`| 320.75 |`
		`+-----------+`

To find the lowest selling price of any type of shoe rounded to 2 decimal places.	`SELECT ROUND(MIN(cost+cost*margin/100),2) AS "Min. SP" FROM shoes;`	`+---------+`
		`| Min. SP |`
		`+---------+`
		`| 135.15 |`
		`+---------+`

AVG() :

AVG() function is used to find the average value of any column or an expression based on a column. AVG() takes one argument which can be any column name or a valid expression involving a column name. Here we have a limitation: the argument of AVG() function can be of numeric (int/decimal) type only. Averages of String and Date type data are not defined. E.g.,

Purpose	Statement	Output

To find the average margin from shoes table.	`SELECT AVG(margin) FROM shoes;`	`+-------------+`
		`| AVG(margin) |`
		`+-------------+`
		`| 2.600000 |`
		`+-------------+`

To find the average cost from the shoes table.	`SELECT AVG(cost) FROM shoes;`	`+------------+`
		`| AVG(cost) |`
		`+------------+`
		`| 491.750000 |`
		`+------------+`

To find the average quantity in stock for the shoes of type Sports.	`SELECT AVG(qty) FROM shoes WHERE type = 'Sports';`	`+----------+`
		`| AVG(qty) |`
		`+----------+`
		`| 580.0000 |`
		`+----------+`

SUM() :

SUM() function is used to find the total value of any column or an expression based on a column. SUM() also takes one argument which can be any column name or a valid expression involving a column name. Like AVG(), the argument of SUM() function can be of numeric (int/decimal) type only. Sums of String and Date type data are not defined. e.g.,

Purpose	Statement	Output

To find the total quantity present in the stock	`SELECT SUM(Qty) FROM Shoes;`	`+----------+`
		`| SUM(Qty) |`
		`+----------+`
		`| 10020 |`
		`+----------+`

To find the total order quantity	`SELECT SUM(order_qty)`
	`FROM orders;`
		`+----------------+`
		`| SUM(order_qty) |`
		`+----------------+`
		`| 2475 |`
		`+----------------+`

To find the the total value (Quanitity x Cost) of Shoes of type 'Office' present in the inventory	`SELECT SUM(cost*qty)`
	`FROM shoes`
	`WHERE type =`
	`'Office';`
		`+---------------+`
		`| SUM(cost*qty) |`
		`+---------------+`
		`| 772000.00 |`
		`+---------------+`

COUNT() :

COUNT() function is used to count the number of values in a column. COUNT() takes one argument which can be any column name, an expression based on a column, or an asterisk (*). When the argument is a column name or an expression based on a column, COUNT() returns the number of non-NULL values in that column. If the argument is a *, then COUNT() counts the total number of rows satisfying the condition, if any, in the table. e.g.,

Purpose	Statement	Output

To count the total number of records in the table Shoes.	`SELECT COUNT(*) FROM shoes;`	`+----------+`
		`| COUNT(*) |`
		`+----------+`
		`| 13 |`
		`+----------+`

To count the different types of shoes that the factory produces	`SELECT COUNT(distinct type)`
	`FROM shoes;`
		`+----------------------+`
		`| COUNT(distinct type) |`
		`+----------------------+`
		`| 3 |`
		`+----------------------+`

To count the records for which the margin is greater than 2.00	`SELECT COUNT(margin)`
	`FROM shoes`
	`WHERE margin > 2;`
		`+---------------+`
		`| COUNT(margin) |`
		`+---------------+`
		`| 5 |`
		`+---------------+`

To count the number of customers in 'A' category	`SELECT COUNT(*)`
	`FROM customers`
	`WHERE category ='A';`
		`+----------+`
		`| COUNT(*) |`
		`+----------+`
		`| 2 |`
		`+----------+`

To count the number of orders of quantity more than 300	`SELECT COUNT(*)`
	`FROM orders`
	`WHERE order_qty > 300;`
		`+----------+`
		`| COUNT(*) |`
		`+----------+`
		`| 2 |`
		`+----------+`

Aggregate functions and NULL values:

None of the aggregate functions takes NULL into consideration. NULL is simply ignored by all the aggregate functions. For example, the statement:

```
SELECT COUNT(*) FROM shoes;
```

produces the following output:

```
+----------+
| COUNT(*)  |
+----------+
|       13  |
+----------+
```

Indicating that there are 13 records in the Shoes table. Whereas the query:

```
SELECT COUNT(margin) FROM shoes;
```

produces the output:

```
+---------------+
| COUNT(margin)  |
+---------------+
|             10  |
+---------------+
```

This output indicates that there are 10 values in the margin column of Shoes table. This means there are 3 (13-10) NULLs in the margin column.

This feature of aggregate functions ensures that NULLs don't play any role in actual calculations. For example, the following statement:

```
SELECT AVG(margin) FROM shoes;
```

produces the output:

```
+-------------+
| AVG(margin)  |
+-------------+
|    2.600000  |
+-------------+
```

The average margin has been calculated by adding all the 10 non NULL values from the margin column and dividing the sum by 10 and not by 13.

Know more!

There are some more aggregate functions available in MySQL. Try to find out what are those. Also try to use them.

GROUP BY

In practical applications many times there arises a need to get reports based on some groups of data. These groups are based on some column values. For example,

- *The management of the shoe factory may want to know what is the total quantity of shoes of various types. i.e., what is the total quantity of shoes of type School, Office, and Sports each.*
- *The management may also want to know what is the maximum, minimum, and average margin of each type of shoes.*
- *It may also be required to find the total number of customers in each category.*

There are many such requirements.

SQL provides GROUP BY clause to handle all such requirements.

For the above three situations, the statements with GROUP BY clause are given below:

In the first situation we want MySQL to divide all the records of shoes table into different groups based on their type (GROUP BY type) and for each group it should display the type and the corresponding total quantity (SELECT type, SUM(qty)). So the complete statement to do this is:

```
SELECT type, SUM(qty) FROM shoes
    GROUP BY type;
```

G1

and the corresponding output is:

```
+---------+-----------+
| type    | SUM(qty)  |
+---------+-----------+
| Office  |      1100 |
| School  |      7180 |
| Sports  |      1740 |
+---------+-----------+
```

Similarly, for the second situation the statement is:

```
SELECT type, MIN(margin), MAX(margin), AVG(margin)
    FROM shoes GROUP BY type;
```
G2

and the corresponding output is:

```
+--------+-------------+-------------+-------------+
| type   | MIN(margin) | MAX(margin) | AVG(margin) |
+--------+-------------+-------------+-------------+
| Office |        3.00 |        3.00 |    3.000000 |
| School |        2.00 |        2.00 |    2.000000 |
| Sports |        3.50 |        3.50 |    3.500000 |
+--------+-------------+-------------+-------------+
```

In the third situation we want MySQL to divide all the records of Customers table into different groups based on the their Category (GROUP BY Category) and for each group it should display the Category and the corresponding number of records (SELECT Category, COUNT(*)). So the complete statement to do this is:

```
SELECT category, COUNT(*) FROM customers GROUP BY category;
```
G3

```
+----------+----------+
| category | COUNT(*) |
+----------+----------+
| A        |        2 |
| B        |        2 |
| C        |        1 |
+----------+----------+
```

Let us have some more examples.

Consider the following statement:

```
SELECT cust_code, SUM(order_qty)
    FROM orders GROUP BY cust_code;
```

This statement produces the following output. Try to explain this this output.

```
+-----------+----------------+
| cust_code | SUM(order_qty) |
+-----------+----------------+
| C001      |           1025 |
| C002      |            750 |
| C003      |            150 |
| C004      |            200 |
| C005      |            350 |
+-----------+----------------+
```

Do the same for the following statement also:

```
SELECT shoe_code, SUM(order_qty)
    FROM orders GROUP BY shoe_code;
```

```
+-----------+----------------+
| shoe_code | SUM(order_qty) |
+-----------+----------------+
| 1001      |            200 |
| 1002      |            200 |
| 1011      |            550 |
| 1012      |            250 |
| 1101      |            300 |
| 1102      |            350 |
| 1103      |            225 |
| 1201      |            200 |
| 1203      |            200 |
+-----------+----------------+
```

If you carefully observe these examples, you will find that GROUP BY is always used in conjunction with some aggregate function(s). A SELECT command with GROUP BY clause has a column name and one or more aggregate functions which are applied on that column and grouping is also done on this column only.

HAVING :

Sometimes we do not want to see the whole output produced by a statement with GROUP BY clause. We want to see the output only for those groups which satisfy some condition. It means we want to put some condition on individual groups (and not on individual records). A condition on groups is applied by HAVING clause. As an example reconsider the

statement G1 discussed above. The statement produced three records in the output - one for each group. Suppose, we are interested in viewing only those groups' output for which the total quantity is more than 1500 (SUM(Qty) > 1500). As this condition is applicable to groups and not to individual rows, we use HAVING clause as shown below:

```
SELECT type, SUM(qty) FROM shoes
   GROUP BY type HAVING SUM(qty) > 1500;
```

```
+--------+----------+
| type   | SUM(qty) |
+--------+----------+
| School |     7180 |
| Sports |     1740 |
+--------+----------+
```

Now suppose for G2 we want the report only for those types for which the average margin is more than 2. For this, following is the statement and the corresponding output:

```
SELECT type, SUM(qty) FROM shoes
   GROUP BY type HAVING AVG(margin) >2;
```

```
+--------+----------+
| type   | SUM(qty) |
+--------+----------+
| Office |     1100 |
| Sports |     1740 |
+--------+----------+
```

In these statements if we try to put the condition using WHERE instead of HAVING, we shall get an error. Another way of remembering this is that whenever a condition involves an aggregate function, then we use HAVING clause in conjunction with GROUP BY clause.

Situations may also arise when we want to put the conditions on individual records as well as on groups. In such situations we use both WHERE (for individual records) and HAVING (for groups) clauses. This can be explained with the help of the following examples:

- The management of the shoe factory may want to know what is the total quantity of shoes, of sizes other than 6, of various types. i.e., what is the total quantity of shoes (of sizes other than 6) of type School, Office, and Sports each.

Moreover, the report is required only for those groups for which the total quantity is more than 1500.

- The management may also want to know what is the maximum, minimum, and average margin of each type of shoes. But in this reports shoes of sizes 6 and 7 only should be included. Report is required only for those groups for which the minimum margin is more than 2.

The statements and their outputs corresponding to above requirements are given below:

```
SELECT type, SUM(qty) FROM shoes
    WHERE size <> 6 <---------- Checks individual row
    GROUP BY type HAVING sum (qty) > 1500; <---- Checks individual group
```

```
+--------+----------+
| type   | SUM(qty) |
+--------+----------+
| School |     3780 |
+--------+----------+
```

```
SELECT type, MIN(margin), MAX(margin), AVG(margin) FROM shoes
    WHERE size in (6,7)
    GROUP BY type having MIN(margin) > 2;
```

```
+--------+-------------+-------------+-------------+
| type   | MIN(margin) | MAX(margin) | AVG(margin) |
+--------+-------------+-------------+-------------+
| Office |        3.00 |        3.00 |    3.000000 |
| Sports |        3.50 |        3.50 |    3.500000 |
+--------+-------------+-------------+-------------+
```

Displaying Data from Multiple Tables

In each situation that we have faced so far, the data was extracted from a single table. There was no need to refer to more than one tables in the same statement. But many times, in real applications of databases, it is required to produce reports which need data from more than one tables. To understand this consider the following situations:

- *The management of the shoe factory wants a report of orders which lists three columns: Order_No, corresponding customer name, and phone number. - (MT-1)*

  *In this case order number will be taken from **Orders** table and corresponding customer name from **Customers** table.*

- *The management wants a four-column report containing order_no, order_qty, name of the corresponding shoe and its cost. - (MT-2)*

  *In this case order number and order quantity will be taken from **Orders** table and corresponding shoe name and cost from **Shoes** table.*

- *The management wants the names of customers who have placed any order of quantity more than 300. - (MT-3)*

  *In this case Order quantity will be checked in **Orders** table and for each record with quantity more than 300, corresponding **Customer** name will be taken from Customers table.*

- *The management wants a report in which with each Order_No management needs name of the corresponding customer and also the total cost (Order quantity x Cost of the shoe) of the order are shown. - (MT-4)*

  *In this case order number will be taken from **Orders** table and corresponding customer name from **Customers** table. For the cost of each order the quantity will be taken from Orders table and the Cost from **Shoes** table.*

In all these cases, the data is to be retrieved from multiple tables. SQL allows us to write statements which retrieve data from multiple tables.

To understand how this is done, consider the following tables of a database.

Product

Code	Name
P001	Toothpaste
P002	Shampoo
P003	Conditioner

Supplier

Sup_Code	Name	Address
S001	DC & Company	Uttam Nagar
S002	SURY Traders	Model Town

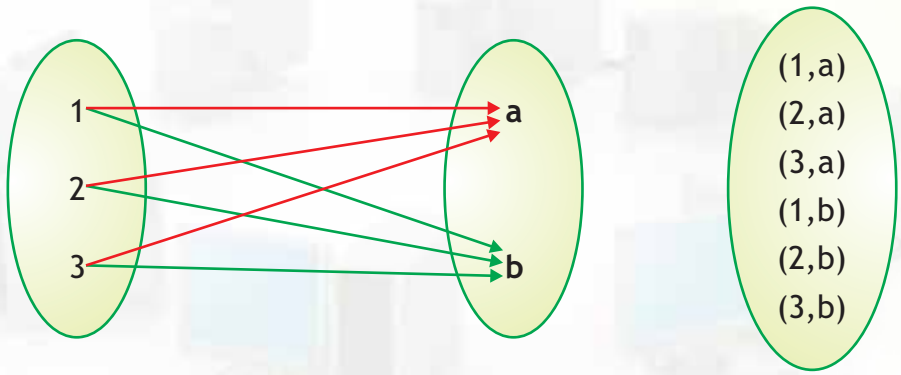
Order_table

Order_No	P_Code	Sup_Code
1	P001	S002
2	P002	S002

These tables are taken just to explain the current concept.

Cartesian Product or Cross Join of tables :

Cartesian product (also called Cross Join) of two tables is a table obtained by pairing up each row of one table with each row of the other table. This way if two tables contain 3 rows and 2 rows respectively, then their Cartesian product will contain 6 (=3x2) rows. This can be illustrated as follows:



Cartesian product of two tables

Notice that the arrows indicate the 'ordered pairing'.

The number of columns in the Cartesian product is the sum of the number of columns in both the tables. In SQL, Cartesian product of two rows is obtained by giving the names of both tables in FROM clause. An example of Cartesian product is shown below:

```
SELECT * FROM order_table, product;
```

To give the output of this query, MySQL will pair the rows of the mentioned tables as follows:

Order_table

Order_No	P_Code	Sup_Code
1	P001	S002
2	P002	S002

Product

Code	Name
P001	Toothpaste
P002	Shampoo
P003	Conditioner

And the following output will be produced:

Order_No	P_Code	Sup_Code	Code	Name	-(CP-1)
1	P001	S002	P001	Toothpaste	
2	P002	S002	P001	Toothpaste	
1	P001	S002	P002	Shampoo	
2	P002	S002	P002	Shampoo	
1	P001	S002	P003	Conditioner	
2	P002	S002	P003	Conditioner	

Here we observe that the Cartesian product contains all the columns from both tables. Each row of the first table (Order_table) is paired with each row of the second table (Product).

If we change the sequence of table names in the FROM clause, the result will remain the same but the sequence of rows and columns will change. This can be observed in the following statement and the corresponding output.

```
SELECT * FROM product, order_table;
```

```
+------+-------------+----------+--------+----------+
| Code | Name        | Order_No | P_Code | Sup_Code |
+------+-------------+----------+--------+----------+
| P001 | Toothpaste  |        1 | P001   | S002     |
| P001 | Toothpaste  |        2 | P002   | S002     |
| P002 | Shampoo     |        1 | P001   | S002     |
| P002 | Shampoo     |        2 | P002   | S002     |
| P003 | Conditioner |        1 | P001   | S002     |
| P003 | Conditioner |        2 | P002   | S002     |
+------+-------------+----------+--------+----------+
```

-(CP-2)

We can have Cartesian product of more than two tables also. Following is the Cartesian Product of three tables:

```
SELECT * FROM order_table, supplier, product;
```

-(CP-3)

```
+----------+--------+----------+----------+-------------+-------------+------+-------------+
| Order_No | P_Code | Sup_Code | Sup_Code | Name        | Address     | Code | Name        |
+----------+--------+----------+----------+-------------+-------------+------+-------------+
|        1 | P001   | S002     | S001     | DC & Company| Uttam Nagar | P001 | Toothpaste  |
|        2 | P002   | S002     | S001     | DC & Company| Uttam Nagar | P001 | Toothpaste  |
|        1 | P001   | S002     | S002     | SURY Traders| Model Town  | P001 | Toothpaste  |
|        2 | P002   | S002     | S002     | SURY Traders| Model Town  | P001 | Toothpaste  |
|        1 | P001   | S002     | S001     | DC & Company| Uttam Nagar | P002 | Shampoo     |
|        2 | P002   | S002     | S001     | DC & Company| Uttam Nagar | P002 | Shampoo     |
|        1 | P001   | S002     | S002     | SURY Traders| Model Town  | P002 | Shampoo     |
|        2 | P002   | S002     | S002     | SURY Traders| Model Town  | P002 | Shampoo     |
|        1 | P001   | S002     | S001     | DC & Company| Uttam Nagar | P003 | Conditioner |
|        2 | P002   | S002     | S001     | DC & Company| Uttam Nagar | P003 | Conditioner |
|        1 | P001   | S002     | S002     | SURY Traders| Model Town  | P003 | Conditioner |
|        2 | P002   | S002     | S002     | SURY Traders| Model Town  | P003 | Conditioner |
+----------+--------+----------+----------+-------------+-------------+------+-------------+
```

The complete Cartesian product of two or more tables is, generally, not used directly. But, some times it is required. Suppose the company with the above database wants to send information of each of its products to each of its suppliers. For follow-up, the management wants a complete list in which each Supplier's detail is paired with each Product's detail. For this, the computer department can produce a list which is the Cartesian product of Product and Supplier tables, as follows:

```
SELECT *, '          ' AS Remarks FROM Product, Supplier;
```

to get the following report:

```
+------+-------------+----------+--------------+-------------+----------+
| Code | Name        | Sup_Code | Name         | Address     | Remarks  |
+------+-------------+----------+--------------+-------------+----------+
| P001 | Toothpaste  | S001     | DC & Company | Uttam Nagar |          |
| P001 | Toothpaste  | S002     | SURY Traders | Model Town  |          |
| P002 | Shampoo     | S001     | DC & Company | Uttam Nagar |          |
| P002 | Shampoo     | S002     | SURY Traders | Model Town  |          |
| P003 | Conditioner | S001     | DC & Company | Uttam Nagar |          |
| P003 | Conditioner | S002     | SURY Traders | Model Town  |          |
+------+-------------+----------+--------------+-------------+----------+
```

Equi- Join of tables :

The complete Cartesian product of two or more tables is, generally, not used directly. Sometimes the complete Cartesian product of two tables may give some confusing information also. For example, the first Cartesian product (CP-1) indicates that each order (Order Numbers 1 and 2) is placed for each Product (Code 'P001', 'P002', 'P003'). But this is incorrect!

Similar is the case with CP-2 and CP-3 also.

But we can extract meaningful information from the Cartesian product by placing some conditions in the statement. For example, to find out the product details corresponding to each Order details, we can enter the following statement:

```
SELECT * FROM order_table, product WHERE p_code = code;

+----------+--------+----------+------+-------------+
| Order_No | P_Code | Sup_Code | Code | Name        |
+----------+--------+----------+------+-------------+
|        1 | P001   | S002     | P001 | Toothpaste  |
|        2 | P002   | S002     | P002 | Shampoo     |
+----------+--------+----------+------+-------------+
```

Two table names are specified in the FROM clause of this statement, therefore MySQL creates a Cartesian product of the tables. From this Cartesian product MySQL selects only those records for which P_Code (Product code specified in the Order_table table) matches Code (Product code in the Product table). These selected records are then displayed.

It always happens that whenever we have to get the data from more than one tables, there is some common column based on which the meaningful data is extracted from the tables. We specify table names in the FROM clause of SELECT command. We also give the condition specifying the matching of common column. (When we say common column, it does not mean that the column names have to be the same. It means that the columns should represent the same data with the same data types.) Corresponding to this statement, internally the Cartesian product of the tables is made. Then based on the specified condition the meaningful data is extracted from this Cartesian product and displayed.

Let us take another example of producing a report which displays the supplier name and address corresponding to each order.

```
SELECT Order_No, Order_table.Sup_Code, Name, Address
    FROM order_table, supplier
    WHERE order_table.sup_code = supplier.sup_code;
```

```
+----------+----------+--------------+------------+
| Order_No | Sup_Code | Name         | Address    |
+----------+----------+--------------+------------+
|        1 | S002     | SURY Traders | Model Town |
|        2 | S002     | SURY Traders | Model Town |
+----------+----------+--------------+------------+
```

In this statement the tables referred are Order_table and Supplier. In these tables sup_code is the common column. This column exists with same name in both the tables. Therefore whenever we mention it, we have to specify the table from which we want to extract this column. This is known as qualifying the column name. If we don't qualify the common column name, the statement would result into an error due to the ambiguous the column names.

Following is another example of equi-join. This time with three tables.

```
Select Order_no, Product.name as Product, Supplier.Name as Supplier
    From order_table, Product, Supplier
        WHERE order_table.Sup_Code = Supplier.Sup_Code
        and P_Code = Code;
```

The output produced by this statement is:

```
+----------+------------+--------------+
| Order_no | Product    | Supplier     |
+----------+------------+--------------+
|        1 | Toothpaste | SURY Traders |
|        2 | Shampoo    | SURY Traders |
+----------+------------+--------------+
```

Let us now get back to our original Shoe database and see how Ms. Akhtar uses the concept of joins to extract data from multiple tables.

For the situation MT-1, she writes the query:

```
SELECT order_no , name, phone
    FROM orders, customers
    WHERE orders.cust_code = customers.cust_code;
```

and get the following required output:

```
+----------+----------------+--------------------+
| order_no | name           | phone              |
+----------+----------------+--------------------+
|        1 | Novelty Shoes  | 4543556, 97878989  |
|        2 | Novelty Shoes  | 4543556, 97878989  |
|        5 | Novelty Shoes  | 4543556, 97878989  |
|        9 | Novelty Shoes  | 4543556, 97878989  |
|        4 | Aaram Footwear | NULL               |
|        6 | Aaram Footwear | NULL               |
|       10 | Aaram Footwear | NULL               |
|        3 | Foot Comfort   | 51917142, 76877888 |
|        7 | Pooja Shoes    | 61345432, 98178989 |
|        8 | Dev Shoes      | NULL               |
+----------+----------------+--------------------+
```

Following are the queries and corresponding outputs for the situations MT-2, MT-3, and MT-4 respectively:

```
SELECT order_no , Order_Qty, name, cost
    FROM orders, shoes WHERE Shoe_Code = code;
```

```
+----------+-----------+----------------+--------+
| order_no | Order_Qty | name           | cost   |
+----------+-----------+----------------+--------+
|        1 |       200 | School Canvas  | 132.50 |
|        2 |       200 | School Canvas  | 135.50 |
|        3 |       150 | School Leather | 232.50 |
|        4 |       250 | School Leather | 270.00 |
|        5 |       400 | School Leather | 232.50 |
|        6 |       300 | Galaxy         | 640.00 |
|        7 |       200 | Tracker        | 700.00 |
|        8 |       350 | Galaxy         | 712.00 |
|        9 |       225 | Galaxy         | 720.00 |
|       10 |       200 | Tracker        | 800.50 |
+----------+-----------+----------------+--------+
```

```
SELECT name, address FROM orders, customers
WHERE orders.cust_code = customers.cust_code
and order_qty > 300;
```

```
+---------------+------------------------+
| name          | address                |
+---------------+------------------------+
| Novelty Shoes | Raja Nagar, Bhopal     |
| Dev Shoes     | Mohan Nagar, Ghaziabad |
+---------------+------------------------+
```

```
SELECT order_no, Order_Qty, customers.name,
    cost*order_qty as 'Order Cost'
    FROM orders, shoes, Customers
    WHERE Shoe_Code = code
          and Orders.Cust_Code = Customers.Cust_Code
    order by order_no;
```

order_no	Order_Qty	name	Order Cost
1	200	Novelty Shoes	26500.00
2	200	Novelty Shoes	27100.00
3	150	Foot Comfort	34875.00
4	250	Aaram Footwear	67500.00
5	400	Novelty Shoes	93000.00
6	300	Aaram Footwear	192000.00
7	200	Pooja Shoes	140000.00
8	350	Dev Shoes	249200.00
9	225	Novelty Shoes	162000.00
10	200	Aaram Footwear	160100.00

Here is another statement extracting data from multiple tables. Try to find out what will be its output and then try this statement on computer and check whether you thought of the correct output.

```
SELECT order_no , Order_Qty, name, cost
    FROM orders, shoes
    WHERE Shoe_Code = code and order_qty > 200;
```

Foreign Key :

As we have just seen, in a join the data is retrieved from the Cartesian product of two tables by giving a condition of equality of two corresponding columns - one from each table. Generally, this column is the Primary Key of one table. In the other table this column is the Foreign key. Such a join which is obtained by putting a condition of equality on cross join is called an 'equi-join'. As an example, once again consider the Product, Supplier, and Order tables referenced earlier. For quick reference these tables are shown once again:

Product

Code	Name
P001	Toothpaste
P002	Shampoo
P003	Conditioner

Supplier

Sup_Code	Name	Address
S001	DC & Company	Uttam Nagar
S002	SURY Traders	Model Town

Order_table

Order_No	P_Code	Sup_Code
1	P001	S002
2	P002	S002

In these tables there is a common column between Product and Order_table tables (Code and P_Code respectively) which is used to get the Equi-Join of these two tables. Code is the Primary Key of Product table and in Order_table table it is not so (we can place more than one orders for the same product). In the order_table, P_Code is a Foreign Key. Similarly, Sup_Code is the primary key in Supplier table whereas it is a Foreign Key is Order_table table. A foreign key in a table is used to ensure referential integrity and to get Equi-Join of two tables.

Referential Integrity: Suppose while entering data in Order_table we enter a P_Code that does not exist in the Product table. It means we have placed an order for an item that does not exist! We should and can always avoid such human errors. Such errors are avoided by explicitly making P_Code a foreign key of Order_table table which always references the Product table to make sure that a non-existing product code is not entered in the Order_table table. Similarly, we can also make Sup_Code a Foreign key in Order_table table which always references Customer table to check validity of Cust_code. This can be done, but how to do it is beyond the scope of this book.

This property of a relational database which ensures that no entry in a foreign key column of a table can be made unless it matches a primary key value in the corresponding related table is called Referential Integrity.

Union

Union is an operation of combining the output of two SELECT statements. Union of two SELECT statements can be performed only if their outputs contain same number of columns and data types of corresponding columns are also the same. The syntax of UNION in its simplest form is:

```
SELECT <select_list>
    FROM <tablename>
    [WHERE <condition> ]
 UNION [ALL]
SELECT <select_list>
    FROM <tablename>
    [WHERE <condition> ];
```

Union does not display any duplicate rows unless ALL is specified with it.

Example:

Suppose a company deals in two different categories of items. Each category contains a number of items and for each category there are different customers. In the database there are two customer tables: Customer_Cat_1 and Customer_Cat_2. If it is required to produce a combined list of all the customers, then it can be done as follows:

```
SELECT Cust_Code from Customer_Cat_1
UNION
SELECT Cust_Code from Customer_Cat_2;
```

If a customer exists with same customer code in both the tables, its code will be displayed only once - because Union does display duplicate rows. If we explicitly want the duplicate rows, then we can enter the statement:

```
SELECT Cust_Code from Customer_Cat_1
UNION ALL
SELECT Cust_Code from Customer_Cat_2;
```

Constraints

Many times it is not possible to keep a manual check on the data that is going into the tables using INSERT or UPDATE commands. The data entered may be invalid. MySQL provides some rules, called Constraints, which help us, to some extent, ensure validity of the data. These constraints are:

S.No.	Constraint	Purpose
1.	PRIMARY KEY	Sets a column or a group of columns as the Primary Key of a table. Therefore, NULLs and Duplicate values in this column are not accepted.
2.	NOT NULL	Makes sure that NULLs are not accepted in the specified column.
3.	FOREIGN KEY	Data will be accepted in this column, if same data value exists in a column in another related table. This other related table name and column name are specified while creating the foreign key constraint.
4.	UNIQUE	Makes sure that duplicate values in the specified column are not accepted.
5.	ENUM	Defines a set of values as the column domain. So any value in this column will be from the specified values only.
6.	SET	Defines a set of values as the column domain. Any value in this column will be a seubset of the specied set only.

We shall discuss only the PRIMARY KEY and NOT NULL constraints in this book. Other constraints are beyond the scope of this book.

PRIMARY KEY:

Recall that primary key of a table is a column or a group of columns that uniquely identifies a row of the table. Therefore no two rows of a table can have the same primary key value. Now suppose that the table Shoes is created with the following statement:

```
CREATE TABLE Shoes
    (Code CHAR(4), Name VARCHAR(20), type VARCHAR(10),
    size INT(2), cost DECIMAL(6,2), margin DECIMAL(4,2),
    Qty INT(4));
```

We know that in this table Code is the Primary key. But, MySQL does not know that! Therefore it is possible to enter duplicate values in this column or to enter NULLs in this column. Both these situations are unacceptable.

To make sure that such data is not accepted by MySQL, we can set Code as the primary key of Shoes table. It can be done by using the PRIMARY KEY clause at the time of table creation as follows:

```
CREATE TABLE Shoes
    (Code CHAR(4) PRIMARY KEY, Name VARCHAR(20),
    type VARCHAR(10), size INT(2), cost DECIMAL(6,2),
    margin DECIMAL(4,2), Qty INT(4));
```

or as follows:

```
CREATE TABLE Shoes
    (Code CHAR(4), Name VARCHAR(20), type VARCHAR(10),
    size INT(2), cost DECIMAL(6,2), margin DECIMAL(4,2),
    Qty INT(4), PRIMARY KEY (Code));
```

To create a table Bills with the combination of columns Order_No and Cust_Code as the primary key, we enter the statement:

```
CREATE TABLE bills
    (Order_Num INT(4) PRIMARY KEY,
    cust_code VARCHAR(4) PRIMARY KEY,
    bill_Date DATE, Bill_Amt DECIMAL(8,2));
```

Contrary to our expectation, we get an error (Multiple primary key defined) with this statement. The reason is that MySQL interprets this statement as if we are trying to create two primary keys of the table - Order_Num, and Cust_code. But a table can have at most one primary key. To set this combination of columns a primary key we have to enter the statement as follows:

```
CREATE TABLE bills
    (Order_Num INT(4), cust_code VARCHAR(4),
    bill_Date date, Bill_Amt DECIMAL(8,2),
    PRIMARY KEY(Order_Num, cust_code));
```

Let us now check the table structure with the command: DESC bills;

The table structure is as shown below:

```
+-----------+-------------+------+-----+---------+-------+
| Field     | Type        | Null | Key | Default | Extra |
+-----------+-------------+------+-----+---------+-------+
| Order_Num | INT(4)      | NO   | PRI | 0       |       |
| cust_code | VARCHAR(4)  | NO   | PRI |         |       |
| bill_Date | date        | YES  |     | NULL    |       |
| Bill_Amt  | DECIMAL(8,2)| YES  |     | NULL    |       |
+-----------+-------------+------+-----+---------+-------+
```

These columns constitute the primary key of the table

NULLs cannot be accepted in these columns.

NOT NULL:

Many times there are some columns of a table in which NULL values should not be accepted. We always want some known valid data values in these columns. For example, we cannot have an order for which the customer code is not known. It means whenever we enter a row in the orders table, corresponding customer code cannot be NULL. Similarly while entering records in the Shoes table, we have to mention the Shoe size, it cannot be set NULL. There may be any number of such situations.

While creating a table we can specify in which columns NULLs should not be accepted as follows:

```
CREATE TABLE Shoes
    (Code CHAR(4) PRIMARY KEY, Name VARCHAR(20),
    type VARCHAR(10), size INT(2) NOT NULL,
    cost DECIMAL(6,2), margin DECIMAL(4,2), Qty INT(4));
CREATE TABLE bills
    (Order_Num INT(4), cust_code VARCHAR(4),
    bill_Date DATE, Bill_Amt DECIMAL(8,2) NOT NULL,
    PRIMARY KEY (Order_Num, cust_code));
```

Now if we try to enter a NULL in the specified column, MySQL will reject the entry and give an error.

Viewing Constraints, Viewing the Columns Associated with Constraints :

After creating a table, we can view its structure using DESC command. The table structure also includes the constraints, if any. Therefore, when we use DESC command, we are shown the table structure as well as constraints, if any. A constraint is shown beside the column name on which it is applicable. E.g., the statement:

```
DESC Shoes;
```

displays the table structure as follows:

```
+--------+--------------+------+-----+---------+-------+
| Field  | Type         | Null | Key | Default | Extra |
+--------+--------------+------+-----+---------+-------+
| Code   | CHAR(4)      | NO   | PRI | NULL    |       |
| Name   | VARCHAR(20)  | YES  |     |         |       |
| type   | VARCHAR(10)  | YES  |     | NULL    |       |
| size   | INT(2)       | NO   |     | 0       |       |
| cost   | DECIMAL(6,2) | YES  |     | NULL    |       |
| margin | DECIMAL(4,2) | YES  |     | NULL    |       |
| Qty    | INT(4)       | YES  |     | NULL    |       |
+--------+--------------+------+-----+---------+-------+
```

ALTER TABLE

In class XI, we have studied that a new column can be added to a table using ALTER TABLE command. Now we shall study how ALTER TABLE can be used:

- to add a constraint
- to remove a constraint
- to remove a column from a table
- to modify a table column

Add, Modify, and Remove constraints :

If we create a table without specifying any primary key, we can still specify its primary key by ALTER TABLE command. Suppose we have created the Shoes table without specifying any Primary key, then later we can enter the statement as follows:

```
ALTER TABLE Shoe ADD PRIMARY KEY(code);
```

This will set Code as the primary key of the table. But if the Code column already contains some duplicate values, then this statement will give an error.

In MySQL, it is also possible to change the primary key column(s) of a table. Suppose, in the Shoes table, istread of Code, we want to set the combination of 'Name' and 'Size' as the primary key. For this first we have to DROP the already existing primary key (i.e., Code) and then add the new primary key (i.e., Name and Size). The corresponding statements are as follows:

```
ALTER TABLE Shoes DROP PRIMARY KEY;
```

After this statement, there is no primary key of Shoe table. Now we can add the new primary key as follows:

```
ALTER TABLE Shoe ADD PRIMARY KEY (Name, Size);
```

Now if we see the table structure by DESC Shoes; statement, it will be shown as follows:

```
+--------+--------------+------+-----+---------+-------+
| Field  | Type         | Null | Key | Default | Extra |
+--------+--------------+------+-----+---------+-------+
| Code   | CHAR(4)      | NO   |     | NULL    |       |
| Name   | VARCHAR(20)  | NO   | PRI |         |       |
| type   | VARCHAR(10)  | YES  |     | NULL    |       |
| size   | INT(2)       | NO   | PRI | 0       |       |
| cost   | DECIMAL(6,2) | YES  |     | NULL    |       |
| margin | DECIMAL(4,2) | YES  |     | NULL    |       |
| Qty    | INT(4)       | YES  |     | NULL    |       |
+--------+--------------+------+-----+---------+-------+
```

In MySQL, it is not possible to add or drop NOT NULL constraint explicitly after the table creation. But it can be done using MODIFY clause of ALTER TABLE command. As an example, suppose we don't want to accept NULL values in bill_date column of bills table, we can issue the statement:

```
ALTER TABLE bills MODIFY bill_date DATE NOT NULL;
```

Later on if we wish to change this status again, we can do so by entering the command:

```
ALTER TABLE bills MODIFY bill_date DATE NULL;
```

Remove and Modify columns :

ALTER TABLE can be used to remove a column from a table. This is done using DROP clause in ALTER TABLE command. The syntax is as follws:

```
ALTER TABLE <tablename> DROP <columnname>
    [, DROP <columnname> [, DROP <columnname> [, . . . ]]];
```

Following are some self explanatory examples of SQL statemenets to remove columns from tables:

```
ALTER TABLE Shoes DROP Qty;

ALTER TABLE Orders DROP Cust_Code;

ALTER TABLE Student DROP Class, DROP RNo, DROP Section;
```

Although any column of a table can be removed, MySQL puts the restriction that a primary key column can be removed only if the remaining, primary key columns, if any, do not contain any duplicate entry. This can be understood more clearly with the help of following example:

The Name and Size columns of the Shoe table constitute its primary key. Now if we drop the Name column from the table, Size will be the remaining Primary Key column of the table. Therefore, duplicate entries in the Size column should not be allowed. To ensure this, before removing Name column from the table, MySQL checks that there are no duplicate entries present in the Size column of the table. If there are any, then the statement trying to remove Name column from the table will result in an error and the Name column will not be removed. If there are no duplicate enteries in the Size column, then Name column will be removed. Similar will be the case with the Name column, if we try to remove Size column. But there won't be any problem if we try to remove both the primary key columns simultaneously with one ALTER TABLE statement as follows:

```
ALTER TABLE Shoes DROP name, DROP size;
```

ALTER TABLE can also be used to change the data type of a table column. For this the syntax is as follows:

```
ALTER TABLE <tablename> MODIFY <col_name> <new datatype>
    [,MODIFY <col_name> <new datatype>
        [,MODIFY <col_name> <new data type>  [, . . . ]]];
```

e.g., the statement:

```
ALTER TABLE shoes modify code CHAR(5), modify type VARCHAR(20);
```

changes the data type of column Code to CHAR(5) and that of type to VARCHAR(20).

When we give a statement to chage the data type of a column, MySQL executes that statement correctly only if the change in data type does not lead to any data loss. E.g., if we try to change the data type of order_date column of orders table from date to int, we'll get an error. This is because the data already stored in this column cannot be converted into int type. Similarly, if a column of VARCHAR(10) type conatins some data value which is 10 characters long, then the data type of this column cannot be converted to VARCHAR(n), where n is an integer less than 10.

DROP TABLE

Sometimes there is a requirement to remove a table from the database. In such cases we don't want merely to delete the data from the table, but we want to delete the table itself. DROP TABLE command is used for this purpose. The syntax of DROP TABLE command is as follows:

```
DROP TABLE <tablename>;
```

e.g. to remove the table Orders from the database we enter the statement:

```
DROP TABLE Orders;
```

And after this statement orders table is no longer available in the database. It has been removed.

Summary

- **Aggregate or Group functions:** MySQL provides Aggregate or Group functions which work on a number of values of a column/expression and return a single value as the result. Some of the most frequently used. Aggregate functions in MySQL are : MIN(), MAX(), AVG(), SUM(), COUNT().
- **Data Types in aggregate functions:** MIN(), MAX(), and COUNT() work on any type of values - Numeric, Date, or String. AVG(), and SUM() work on only Numeric values (INT and DECIMAL).
- **NULLs in aggregate functions:** Aggregate functions totally ignore NULL values present in a column.
- **GROUP BY:** GROUP BY clause is used in a SELECT statement in conjunction with aggregate functions to group the result based on distinct values in a column.
- **HAVING:** HAVING clause is used in conjuction with GROUP BY clause in a SELECT statement to put condition on groups.
- **WHERE Vs HAVING:** WHERE is used to put a condition on individual row of a table whereas HAVING is used to put condition on individual group formed by GROUP BY clause in a SELECT statement.

- **Cartesian Product (or Cross Join):** Cartesian product of two tables is a table obtained by pairing each row of one table with each row of the other. A cartesian product of two tables contains all the columns of both the tables.
- **Equi-Join:** An equi join of two tables is obtained by putting an equality condition on the Cartesian product of two tables. This equality condition is put on the common column of the tables. This common column is, generally, primary key of one table and foreign key of the other.
- **Foreign Key:** It is a column of a table which is the primary key of another table in the same database. It is used to enforce referential integrity of the data.
- **Referential Integrity:** The property of a relational database which ensures that no entry in a foreign key column of a table can be made unless it matches a primary key value in the corresponding column of the related table.
- **Union:** Union is an operation of combining the output of two SELECT statements.
- **Constraints:** These are the rules which are applied on the columns of tables to ensure data integrity and consistency.
- **ALTER TABLE:** ALTER TABLE command can be used to Add, Remove, and Modify columns of a table. It can also be used to Add and Remove constraints.
- **DROP TABLE:** DROP TABLE command is used to delete tables.

EXERCISES

MULTIPLE CHOICE QUESTIONS

1. Which of the following will give the same answer irrespective of the NULL values in the specified column:

   a. MIN() b. MAX()

   c. SUM() d. None of the above

2. An aggregate function:

   a. Takes a column name as its arguments

   b. May take an expression as its argument

c. Both (a) and (b)

d. None of (a) and (b)

3. HAVING is used in conjunction with

a. WHERE
b. GROUP BY clause
c. Aggregate functions
d. None of the above

4. In the FROM clause of a SELECT statement

a. Multiple Column Names are specified.

b. Multiple table names are specified.

c. Multiple Column Names may be specified.

d. Multiple table names may be specified.

5. JOIN in RDBMS refers to

a. Combination of multiple columns
b. Combination of multiple rows
c. Combination of multiple tables
d. Combination of multiple databases

6. Equi-join is formed by equating

a. Foreign key with Primary key
b. Each row with all other rows
c. Primary key with Primary key
d. Two tables

7. Referential integrity

a. Must be maintained

b. Cannot be maintained

c. Is automatically maintained by databases

d. Should not be maintained

8. A Primary key column

a. Can have NULL values
b. Can have duplicate values
c. Both (a) and (b)
d. Neither (a) nor (b)

9. Primary Key of a table can be
   a. Defined at the time of table creation only.
   b. Defined after table creation only.
   c. Can be changed after table creation
   d. Cannot be changed after table creation
10. Two SELECT commands in a UNION
   a. Should select same number of columns.
   b. Should have different number of columns
   c. Both (a) and (b)
   d. Neither (a) nor (b)

ANSWER THE FOLLOWING QUESTIONS

1. Why are aggregate functions called so? Name some aggregate functions.
2. Why is it not allowed to give String and Date type arguments for SUM() and AVG() functions? Can we give these type of arguments for other functions?
3. How are NULL values treated by aggregate functions?
4. There is a column C1 in a table T1. The following two statements:

   `SELECT COUNT(*) FROM T1;` and `SELECT COUNT(C1) from T1;`

   are giving different outputs. What may be the possible reason?
5. What is the purpose of GROUP BY clause?
6. What is the difference between HAVING and WHERE clauses? Explain with the help of an example.
7. What is the Cartesian product of two table? Is it same as an Equi-join?
8. There are two table T1 and T2 in a database. Cardinality and degree of T1 are 3 and 8 respectively. Cardinality and degree of T2 are 4 and 5 respectively. What will be the degree and Cardinality of their Cartesian product?

9. What is a Foreign key? What is its importance?

10. What are constraints? Are constraints useful or are they hinderance to effective management of databases?

11. In a database there is a table Cabinet. The data entry operator is not able to put NULL in a column of Cabinet? What may be the possible reason(s)?

12. In a database there is a table Cabinet. The data entry operator is not able to put duplicate values in a column of Cabinet? What may be the possible reason(s)?

13. Do Primary Key column(s) of a table accept NULL values?

14. There is a table T1 with combination of columns C1, C2, and C3 as its primary key? Is it possible to enter:

    a. NULL values in any of these columns?

    b. Duplicate values in any of these columns?

15. At the time of creation of table X, the data base administrator specified Y as the Primary key. Later on he realized that instead of Y, the combination of column P and Q should have been the primary key of the table. Based on this scenario, answer the following questions:

    a. Is it possible to keep Y as well as the combination of P and Q as the primary key?

    b. What statement(s) should be entered to change the primary key as per the requirement.

16. Does MySQL allow to change the primary key in all cases? If there is some special case, please mention.

17. What are the differences between DELETE and DROP commands of SQL?

LAB EXERCISES

1. In a database create the following tables with suitable constraints :

STUDENTS

AdmNo	Name	Class	Sec	RNo	Address	Phone
1271	Utkarsh Madaan	12	C	1	C-32, Punjabi Bagh	4356154
1324	Naresh Sharma	10	A	1	31, Mohan Nagar	435654
1325	Md. Yusuf	10	A	2	12/21, Chand Nagar	145654
1328	Sumedha	10	B	23	59, Moti Nagar	4135654
1364	Subya Akhtar	11	B	13	12, Janak Puri	NULL
1434	Varuna	12	B	21	69, Rohini	NULL
1461	David DSouza	11	B	1	D-34, Model Town	243554, 98787665
2324	Satinder Singh	12	C	1	1/2, Gulmohar Park	143654
2328	Peter Jones	10	A	18	21/32B, Vishal Enclave	24356154
2371	Mohini Mehta	11	C	12	37, Raja Garden	435654, 6765787

SPORTS

AdmNo	Game	CoachName	Grade
1324	Cricket	Narendra	A
1364	Volleball	M.P. Singh	A
1271	Volleball	M.P. Singh	B
1434	Basket Ball	I. Malhotra	B
1461	Cricket	Narendra	B
2328	Basket Ball	I. Malhotra	A
2371	Basket Ball	I. Malhotra	A
1271	Basket Ball	I. Malhotra	A
1434	Cricket	Narendra	A
2328	Cricket	Narendra	B
1364	Basket Ball	I. Malhotra	B

a) Based on these tables write SQL statements for the following queries:

i. Display the lowest and the highest classes from the table STUDENTS.

ii. Display the number of students in each class from the table STUDENTS.

iii. Display the number of students in class 10.

iv. Display details of the students of Cricket team.

v. Display the Admission number, name, class, section, and roll number of the students whose grade in Sports table is 'A'.

vi. Display the name and phone number of the students of class 12 who are play some game.

vii. Display the Number of students with each coach.

viii. Display the names and phone numbers of the students whose grade is 'A' and whose coach is Narendra.

b) Identify the Foreign Keys (if any) of these tables. Justify your choices.

c) Predict the the output of each of the following SQL statements, and then verify the output by actually entering these statements:

i. SELECT class, sec, count(*) FROM students GROUP BY class, sec;

ii. SELECT Game, COUNT(*) FROM Sports GROUP BY Game;

iii. SELECT game, name, address FROM students, Sports

WHERE students.admno = sports.admno AND grade = 'A';

iv. SELECT Game FROM students, Sports

WHERE students.admno = sports.admno AND Students.AdmNo = 1434;

2. In a database create the following tables with suitable constraints :

ITEMS

I_Code	Name	Category	Rate
1001	Masala Dosa	South Indian	60
1002	Vada Sambhar	South Indian	40
1003	Idli Sambhar	South Indian	40
2001	Chow Mein	Chinese	80
2002	Dimsum	Chinese	60
2003	Soup	Chinese	50
3001	Pizza	Italian	240
3002	Pasta	Italian	125

BILLS

BillNo	Date	I_Code	qty
1	2010-04-01	1002	2
1	2010-04-01	3001	1
2	2010-04-01	1001	3
2	2010-04-01	1002	1
2	2010-04-01	2003	2
3	2010-04-02	2002	1
4	2010-04-02	2002	4
4	2010-04-02	2003	2
5	2010-04-03	2003	2
5	2010-04-03	3001	1
5	2010-04-03	3002	3

a) Based on these tables write SQL statements for the following queries:

i. Display the average rate of a South Indian item.

ii. Display the number of items in each category.

iii. Display the total quantity sold for each item.

iv. Display total quanity of each item sold but don't display this data for the items whose total quantity sold is less than 3.

v. Display the details of bill records along with Name of each corresponding item.

vi. Display the details of the bill records for which the item is 'Dosa'.

vii. Display the bill records for each Italian item sold.

viii. Display the total value of items sold for each bill.

b) Identify the Foreign Keys (if any) of these tables. Justify your answer.

c) Answer with justification (Think independently. More than one answers may be correct. It all depends on your logical thinking):

i. Is it easy to remember the Category of item with a given item code? Do you find any kind of pattern in the items code? What could be the item code of another South Indian item?

ii. What can be the possible uses of Bills table? Can it be used for some analysis purpose?

iii. Do you find any columns in these tables which can be NULL? Is there any column which must not be NULL?

3. In a database create the following tables with suitable constraints :

VEHICLE

Field	Type	Null	Key	Default	Extra
RegNo	char(10)	NO	PRI		
RegDate	date	YES		NULL	
Owner	varchar(30)	YES		NULL	
Address	varchar(50)	YES		NULL	

CHALLAN

Field	Type	Null	Key	Default	Extra
Challan_No	int(11)	NO	PRI	0	
Ch_Date	date	YES		NULL	
RegNo	char(10)	YES		NULL	
Offence	int(3)	YES		NULL	

OFFENCE

Field	Type	Null	Key	Default	Extra
Offence_Code	int(3)	NO	PRI	0	
Off_desc	varchar(30)	YES		NULL	
Challan_Amt	int(4)	YES		NULL	

a) Based on these tables write SQL statements for the following queries:

i. Display the dates of first registration and last registration from the table Vehicle.

ii. Display the number of challans issued on each date.

iii. Display the total number of challans issued for each offence.

iv. Display the total number of vehicles for which the 3rd and 4th characters of RegNo are '6C'.

v. Display the total value of challans issued for which the Off_Desc is 'Driving without License'.

vi. Display details of the challans issued on '2010-04-03' along with Off_Desc for each challan.

vii. Display the RegNo of all vehicles which have been challaned more than once.

viii. Display details of each challan alongwith vehicle details, Off_desc, and Challan_Amt.

b) Identify the Foreign Keys (if any) of these tables. Justify your choices.

c) Should any of these tables have some more column(s)? Think, discuss in peer groups, and discuss with your teacher.

4. In a database create the following tables with suitable constraints:

Table: Employee

No	Name	Salary	Zone	Age	Grade	Dept
1	Mukul	30000	West	28	A	10
2	Kritika	35000	Centre	30	A	10
3	Naveen	32000	West	40		20
4	Uday	38000	North	38	C	30
5	Nupur	32000	East	26		20
6	Moksh	37000	South	28	B	10
7	Shelly	36000	North	26	A	30

Table: Department

Dept	DName	MinSal	MaxSal	HOD
10	Sales	25000	32000	1
20	Finance	30000	50000	5
30	Admin	25000	40000	7

a) Based on these tables write SQL statements for the following queries:

i. Display the details of all the employees who work in Sales department.

ii. Display the Salary, Zone, and Grade of all the employees whose HOD is Nupur.

iii. Display the Name and Department Name of all the employees.

iv. Display the names of all the employees whose salary is not within the specified range for the corresponding department.

v. Display the name of the department and the name of the corresponding HOD for all the departments.

b) Identify the Foreign Keys (if any) of these tables. Justify your choices.

TEAM BASED TIME BOUND EXERCISE:

(Team size recommended: 3 students each team)

1. A chemist shop sells medicines manufactured by various pharmaceutical companies. When some medicine is sold, the corresponding stock decreases and when some medicines are bought (by the chemist shop) from their suppliers, the corresponding stock increases. Now the shop wants to keep computerized track of its inventory. The shop owner should be able to find

- The current stock of any medicine.
- The total sale amount of any specific time period (a specific day, or month, or any period between two specific dates)
- The details of all the medicines from a specific supplier.

- The details of all the medicines from a specific manufacturer.
- Total value of the medicines in the stock.

There may be a number of other reports which the shop owner may like to have.

The job of each team is to design a database for this purpose. Each team has to specify:

- The structure (with constraints) of each of the tables designed (with justification).
- How the tables are related to each other (foreign keys).
- How the design will fulfill all the mentioned requirements.
- At least 10 reports that can be generated with the database designed.

2. To expand its business, XYZ Mall plans to go online. Anyone who shops at the Mall will be given a membership number and Password which can be used for online shopping. With this membership number and password, customers can place their orders online. The mall will maintain the customers' data and orders' data. A person is put on duty to keep constantly checking the Orders data. Whenever an order is received, its processing has to start at the earliest possible.

   The Orders' data will be analysed periodically (monthly, quarterly, annually - whatever is suitable) to further improve business and customer satisfaction.

   The job of each team is to design a database for this purpose. Each team has to specify:

   - The structure (with constraints) of each of the tables designed (with justification).
   - How the tables are related to each other (foreign keys).
   - How the design will fulfill all the mentioned requirements.
   - At least 10 reports that can be generated with the database designed.

ADVANCED RDBMS CONCEPTS

Learning Objectives

After studying this lesson the students will be able to:

- Define a Transaction
- Describe reason why all the tasks in a transaction should be executed fully or not at all.
- Perform basic transactions.
- Commit a transaction.
- Add Save Points to a transaction.
- Roll back a Transaction
- Roll back a Transaction to a Savepoint.

Till now we have studied about various SQL statements manipulating data stored in a MySQL database. We executed SQL statements without concern about inconsistencies arising due to group of statements not being executed in entirety. In this lesson, we will study the basic concepts of Transaction processing and how MySQL ensures consistency of data when a group of statements is executed.

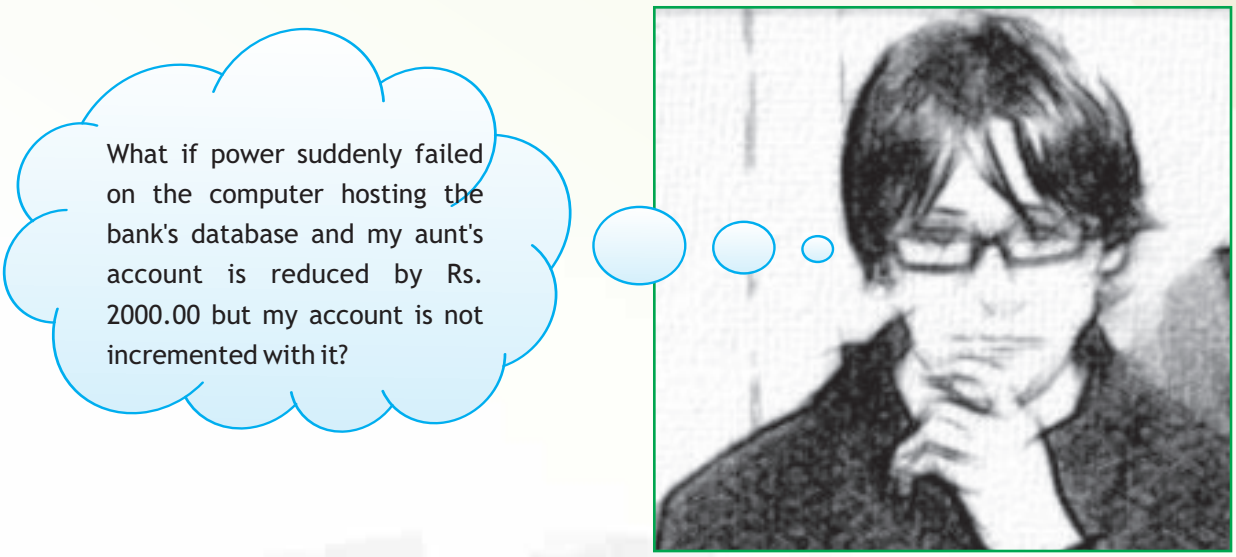
Puzzle [9]

Vijaya has to withdraw ₹ 2500.00 from her account in the bank. She asked for m notes of ₹ 50.00 and n notes of ₹ 100.00. The cashier made a mistake and handed her m notes of ₹ 100.00 and n notes of ₹ 50.00. When she returned back home she realized that she got ₹ 500.00 less. How many notes of ₹ 50.00 and ₹ 100.00 did she ask for?

Introduction

Raunak studies in Class XII. He is very helpful. During summer vacations, he helped his aunt's son in his studies. His aunt was very pleased with him and gave him a cheque of ₹ 2000.00. Raunak knows that after presentation of the cheque to the bank, his aunt's account will be reduced by ₹ 2000.00 and his account will be increased by ₹ 2000.00. Raunak walked up to the bank to present the cheque.

While returning from bank, Raunak is apprehensive about one thing:



Are Raunak's fears valid? Can you recollect something like this happening with you or with your friends or relatives?

Raunak should not fear at all as the Bank's DBMS looks after these eventualities. DBMSs ensure consistency (correctness) of data by managing Transactions.

DBMS and Transaction Management

Mostly customers view an operation like transfer of funds as a single operation but actually it consists of series of operations.

Suppose Raunak's account number is 3246 and his aunt's account number is 5135. In order to process the cheque presented by Raunak, the following two SQL commands need to be executed on the database maintained by the bank:

```
UPDATE Savings
SET balance = balance - 2000        <-- For Aunt's account
WHERE account_no = 5135;

UPDATE Savings
SET balance = balance + 2000        <-- For Raunak's account
WHERE account_no = 3246;
```

The above two Updates should both take place. If the first Update takes place and there is a system failure, the first updation should be undone. Either both the updations should be done and if it is not possible for both the updations to be done, then no updation should be done.

What is a Transaction?

A Transaction is a unit of work that must be done in logical order and successfully as a group or not done at all. Unit of work means that a Transaction consists of different tasks - but together they are considered as one unit. Each transaction has a beginning and an end. If anything goes wrong in between the execution of transaction, the entire transaction (No matter to what extent has been done) should be cancelled. If it is successful, then the entire transaction should be saved to the database.

> A transaction is a unit of work that must be done in logical order and successfully as a group or not done at all.

In Raunak's case, both the updation statements constitute a transaction. Both are together treated as a single unit.

To understand how transactions are managed, let us study the following 3 statements of SQL:

- START TRANSACTION statement
- COMMIT statement
- ROLLBACK statement

START TRANSACTION Statement :

START TRANSACTION statement commits the current transaction and starts a new transaction. It tells MySQL that the transaction is beginning and the statements that follow should be treated as a unit, until the transaction ends. It is written like this:

```
START TRANSACTION;
```

> The **START TRANSACTION** statement has no clauses.

COMMIT Statement :

The COMMIT statement is used to save all changes made to the database during the transaction to the database.Commit statement is issued at a time when the transaction is complete- all the changes have been successful and the changes should be saved to the database. COMMIT ends the current transaction.

COMMIT statement is used like this:

```
COMMIT;
```

Or

```
COMMIT WORK;
```

Here WORK is a keyword and is optional.

In the following example, the table named savings has 2 rows. A transaction is started and balance in Siddharth's account (with account number 1004) is increased by Rs. 2000.00 and the balance in Akriti's account (with account number 1006) is decreased by Rs. 2000.00. COMMIT statement makes the changes made by the transaction permanent.

Example 1:

```
mysql> select * from savings;
+------------+------------------+----------+
| account_no | name             | balance  |
+------------+------------------+----------+
|       1004 | Siddharth Sehgal | 87000.00 |
|       1006 | Akriti Malik     | 87000.00 |
+------------+------------------+----------+
mysql> START TRANSACTION;
mysql> UPDATE Savings
    -> SET balance = balance + 2000
    -> WHERE account_no = 1004;
mysql> UPDATE Savings
    -> SET balance = balance - 2000
    -> WHERE account_no = 1006;
mysql> SELECT * FROM Savings;
+------------+------------------+----------+
| account_no | name             | balance  |
+------------+------------------+----------+
|       1004 | Siddharth Sehgal | 89000.00 |
|       1006 | Akriti Malik     | 85000.00 |
+------------+------------------+----------+
2 rows in set (0.00 sec)
mysql> COMMIT;
```

ROLLBACK Statement :

When a transaction is being executed, some type of error checking is usually performed to check whether it is executing successfully or not. If not, the entire transaction is undone using the ROLLBACK statement. The ROLLBACK statement cancels the entire transaction i.e. It rolls the transaction to the beginning. It aborts any changes made during the transaction and the state of database is returned to what it was before the transaction began to execute and does not save any of the changes made to the database during the transaction.

ROLLBACK statement is used like this:

```
ROLLBACK;
```

Or

```
ROLLBACK WORK;
```

Here WORK is a keyword and is optional.

If in Example 1 shown above ROLLBACK was used instead of COMMIT, the updation of incrementing Siddharth's account by ₹ 2000.00 and decrementing Akriti's account by ₹ 2000 wouldn't have taken place. Let us now initiate a transaction, increase Akriti's account by ₹ 3000.00, then Rollback the transaction and see what happens to the updation done on Akriti's account.

```
mysql> SELECT * FROM Savings;
+------------+------------------+----------+
| account_no | name             | balance  |
+------------+------------------+----------+
|       1004 | Siddharth Sehgal | 89000.00 |
|       1006 | Akriti Malik     | 85000.00 |
+------------+------------------+----------+
mysql> START TRANSACTION;
mysql> UPDATE Savings
```

Before the transaction starts, Siddharth's balance is Rs. 89000 and Akriti's balance is Rs. 85000.00

```
    -> SET balance = balance + 3000
    -> WHERE account_no = 1006;
mysql> ROLLBACK;
mysql> SELECT * FROM Savings;
+------------+------------------+----------+
| account_no | name             | balance  |
+------------+------------------+----------+
|       1004 | Siddharth Sehgal | 89000.00 |
|       1006 | Akriti Malik     | 85000.00 |
+------------+------------------+----------+
```

Akriti's balance is increased by Rs. 3000.00

Because of the Rollback, Akriti's balance is not updated and is displayed as it was before the transaction started.

- After the ROLLBACK command is issued to the database, the database itself starts a new transaction; though no explicit command of starting a transaction like START TRANSACTION is issued.

Example 2:

Let us try out some more SQL statements on Savings table to understand transactions well.

```
mysql> SELECT * FROM savings;
+------------+------------------+----------+
| account_no | name             | balance  |
+------------+------------------+----------+
|       1004 | Siddharth Sehgal | 84000.00 |
|       1006 | Akriti Malik     | 92000.00 |
|       1008 | Chavi Mehra      | 67000.00 |
|       1009 | Raunak Singh     | 56000.00 |
+------------+------------------+----------+
```

```
mysql> INSERT INTO Savings VALUES
    (1010,'Lakshmi Swamy',34000);
mysql> START TRANSACTION;
mysql> UPDATE Savings SET balance =
balance +2000 WHERE account_no = 1010;
mysql> ROLLBACK;
mysql> SELECT * FROM Savings;
+------------+-----------------+----------+
| account_no | name            | balance  |
+------------+-----------------+----------+
|       1004 | Siddharth Sehgal | 84000.00 |
|       1006 | Akriti Malik     | 92000.00 |
|       1008 | Chavi Mehra      | 67000.00 |
|       1009 | Raunak Singh     | 56000.00 |
|       1010 | Lakshmi Swamy    | 34000.00 |
+------------+-----------------+----------+
5 rows in set (0.00 sec)
```

Start transaction statement starts a transaction and commits the previous INSERT INTO statement.

Rollback cancels the effect of Update statement.

SELECT statement displays Lakshmi Swamy's row with balance of 34000.00

Inserting SavePoints :

The SAVEPOINT statement defines a marker in a transaction. These markers are useful in rolling back a transaction till the marker.

We can add a savepoint anywhere in a transaction. When you roll back to that savepoint, any changes made to the database after the savepoint are discarded, and any changes made prior to the savepoint are saved. It is like semicomitting a transaction.

To define a savepoint, we enter the SAVEPOINT statement like this:

```
SAVEPOINT <savepoint-name>;
```

Example :

```
SAVEPOINT Mark1;
```

In the above statement a marker (savepoint) with the name Mark1 is defined. It becomes a bookmark in the transaction. Now we can write the following statement:

```
ROLLBACK TO SAVEPOINT Mark1;
```

to rollback the transaction till the bookmark named Mark1.

Setting Autocommit :

By default, Autocommit mode is on in MySQL. It means that MySQL does a COMMIT after every SQL statement that does not return an error. If it returns an error then either Rollback or Commit happens depending on the type of error. If we do not want individual statements of SQL to be automatically committed, we should set the autocommit mode to off.

When Autocommit is off then we have to issue COMMIT statement explicitly to save changes made to the database.

The following statement sets the autocommit mode to off. It also starts a new transaction

```
SET AUTOCOMMIT=0;
```

The following statement sets the autocommit mode to ON. It also commits and terminates the current transaction.

```
SET AUTOCOMMIT=1;
```

If autocommit is set to ON. we can still perform a multiple-statement transaction by starting it with an explicit START TRANSACTION statement and ending it with COMMIT or ROLLBACK.

Let us look at the following example to understand it:

Example

```
mysql> SET AUTOCOMMIT = 0;
mysql> SELECT * FROM Savings;
+------------+------------------+----------+
| account_no | name             | balance  |
+------------+------------------+----------+
|       1004 | Siddharth Sehgal | 84000.00 |
|       1006 | Akriti Malik     | 92000.00 |
|       1008 | Chavi Mehra      | 67000.00 |
+------------+------------------+----------+
mysql> INSERT INTO Savings values
(1009,'Raunak Singh', 56000);
mysql> ROLLBACK;
mysql> SELECT * FROM Savings;
+------------+------------------+----------+
| account_no | name             | balance  |
+------------+------------------+----------+
|       1004 | Siddharth Sehgal | 84000.00 |
|       1006 | Akriti Malik     | 92000.00 |
|       1008 | Chavi Mehra      | 67000.00 |
+------------+------------------+----------+
mysql> SET AUTOCOMMIT = 1;
mysql> INSERT INTO Savings VALUES
(1009,'Raunak Singh', 56000);
```

Autocommit is disabled.

Table Savings has 3 rows.

Another row for Raunak Singh added.

Insert statement was not committed so it is undone by Rollback

Table does not show Raunak Singh's row.

Autocommit is enabled.

Raunak's row is added and is committed too.

```
mysql> ROLLBACK;
mysql> SELECT * FROM Savings;
+------------+------------------+----------+
| account_no | name             | balance  |
+------------+------------------+----------+
|       1004 | Siddharth Sehgal | 84000.00 |
|       1006 | Akriti Malik     | 92000.00 |
|       1008 | Chavi Mehra      | 67000.00 |
|       1009 | Raunak Singh     | 56000.00 |
+------------+------------------+----------+
```

Rollback cannot undo insertion of Raunak's row.

If the autocommit mode has been set to off in a session and you end that session, the autocommit mode is automatically set to on when you start a new session.

Let us try out some more SQL statements :

Example

```
mysql> SET AUTOCOMMIT = 1;
Query OK, 0 rows affected (0.00 sec)

mysql> START TRANSACTION;
Query OK, 0 rows affected (0.00 sec)

mysql> DELETE FROM Savings WHERE account_no = 1006;
Query OK, 1 row affected (0.00 sec)

mysql> ROLLBACK WORK;
Query OK, 0 rows affected (0.03 sec)
```

Autocommit is enabled

Start transaction sets autocommit off.

Row with account_no 1006 deleted but is not committed.

Deletion of Row with account_no 1006 is cancelled.

An implicit COMMIT takes place, even if AUTOCOMMIT is set OFF, on the database when the user issues a Data Definition language command like CREATE TABLE, ALTER TABLE etc .

Future Trends

Cloud Computing

A major shift is predicted in the IT industry in the way that software and services are going to be delivered in future. In this regard, cloud computing is an emerging area these days. It is the computing model where the infrastructure and the applications are offered as a service over the Internet. Cloud computing takes place out on someone else's network. Since the details of how it is set up or how it works is hidden from the user, the term cloud is used. Cloud means a large network that is away and is not in our control. Just about any database can be run in a cloud-based infrastructure. Search the web and find out how RDBMSs like MySQL will be significant in such a computing model.

Summary

- Work done during a transaction is a series of operations.
- If one of the operations of a transaction is not executed successfully, then the entire transaction should be cancelled. If all the operations are executed successfully, the transaction should be saved to a database.
- START TRANSACTION statement is used to start a transaction.
- The process of cancelling a transaction is called Rolling back.
- ROLLBACK statement is used to terminate a transaction and roll back the database to its original state before the transaction.
- COMMIT statement is used to save changes to the database.
- When AutoCommit is ON, each SQL statement is a transaction. The changes resulting from each statement are automatically committed.

- When Auto Commit is Off then changes made to database are not committed unless explicitly requested.

EXERCISES

MULTIPLE CHOICE QUESTIONS

1. A _____is a logical unit of work that must succeed or fail in its entirety.

   a) Primary key

   b) Database

   c) Transaction

   d) none of these

2. When AutoCommit is _____ ,changes made to database are not committed unless explicitly requested.

   a) Equal to "-"

   b) on

   c) off

   d) Equal to "%"

3. When a CREATE TABLE command is issued, a _____occurs on the database.

   a) ROLLBACK

   b) COMMIT

   c) SAVEPOINT

   d) ROLLBACK TO SAVEPOINT

4. By default AUTOCOMMIT is _____.

   a) disabled

   b) enabled

c) inactive

d) none of the above

5. Which of the following statement or command? Completes a transaction?

a) INSERT INTO

b) COMMIT

c) DELETE

d) SELECT

6. If Feroze deposits a cheque of Rs. 1200.00 in his account, which was given to him by Ali, two tasks: decreasing of Rs. 1200.00 from Ali's account and increment of Rs. 1200.00 in Feroze's account are done. _____ constitute(s) a transaction.

a) First task

b) Both the tasks

c) None of the tasks

d) Second task.

7. START TRANSACTION statement

a) Updates the current database.

b) Rolls back the current transaction.

c) Commits the current transaction and starts a new transaction

d) Starts a new transaction only.

8. ROLLBACK statement

a) Cancels the entire transaction.

b) Deletes the rows added to the table.

c) Commits the current transaction and starts a new transaction

d) Rolls back all the insertions of rows made during the current transaction.

ANSWER THE FOLLOWING QUESTIONS

1. Define a Transaction.
2. Explain with the help of an example that why should a transaction be executed as a whole or it should be not executed at all.
3. Distinguish between COMMIT and ROLLBACK.
4. Write the purpose of inserting Savepoints in a transaction.
5. What happens when Autocommit is set on?
6. Write SQL statement to set Autocommit to off.
7. What does the ROLLBACK statement do?
8. How do you tell the system that a transaction is beginning?
9. Why do we generally execute a COMMIT statement before beginning a transaction?
10. Name two statements that can be used to end a transaction?
11. Does executing the COMMIT or ROLLBACK statement end the current transaction?
12. What happens to the current transaction if a DDL Statement is executed?

LAB EXERCISES

a) Perform the following tasks:

Start MySQL session .

Create a table named Student with columns RollNumber, Name and Marks.

Start a transaction and insert two rows to the Student table.

Verify the inserts by SELECT statement.

Commit the changes.

Start another transaction.

Delete a row that was recently inserted.

Verify that the row has been deleted.

Rollback the changes.

Verify that the delete has been cancelled.

b) A table named ITEM has the following contents:

```
+-------+--------------+----------+
| icode  | iname         | iprice    |
+-------+--------------+----------+
|   101  | CHAIR         |  1500.00  |
|   102  | DINING TABLE  | 24000.00  |
+-------+--------------+----------+
```

Write the output that will be displayed by each SELECT statement as the SQL statements given below are executed:

```
mysql> SELECT * FROM ITEM;
mysql> SET AUTOCOMMIT = 0;
mysql> INSERT INTO ITEM VALUES(103,'COFFEE TABLE',340);
mysql> SELECT * FROM ITEM;
mysql> ROLLBACK;
mysql> SELECT * FROM ITEM;
mysql> START TRANSACTION;
mysql> UPDATE ITEM SET IPRICE = IPRICE +200;
mysql> SAVEPOINT S1;
mysql> UPDATE ITEM SET IPRICE = IPRICE +400;
mysql> SELECT * FROM ITEM;
mysql> ROLLBACK TO  S1;
mysql> SELECT * FROM ITEM;
```

Now verify the output by creating the table using MySQL and executing the above statements.

c) A table named Bill has the following rows:

```
+-----------+-----------+------------+-----------+
| Order_Num | cust_code | bill_date  | Bill_Amt - |
+-----------+-----------+------------+-----------+
|         1 | C101      | 2010-08-02 |     2300  |
|         2 | C105      |  2010-08-02|     5500  |
|         3 | C099      |  2010-08-23|     3000  |
|         4 | C165      |  2010-09-24|      6500 |
|         5 | C105      |  2010-09-24|      1400 |
+-----------+-----------+------------+-----------+
```

Write the output that will be displayed due to last SQL SELECT statement:

```
mysql> START TRANSACTION;
mysql> INSERT INTO BILLS VALUES(7,'C101','2010-09-02',5000);
mysql> UPDATE BILLS SET Bill_Amt = Bill_Amt+500 WHERE
Order_Num = 3;
mysql> SAVEPOINT A;
mysql> INSERT INTO BILLS VALUES(8,'C97','2010-09-03',4500);
mysql> DELETE FROM BILL WHERE cust_code = 'C105';
mysql> ROLLBACK TO A;
mysql> SELECT * FROM bills;
```

Now verify the output by actually executing the statements.

TIME BOUND TEAM BASED EXERCISE

(Team size recommended : 3 students each team)

Suppose the Government has nominated you as Head of Ministry of Information Technology. You have come out with the idea of a National Identity Card that should be

present with each citizen. The card will hold a limited amount of information but will be backed up by further information held in a database with the Government. Some information will be visible on the card, some encoded on chip in the card and other detailed information about each citizen will be present in tables in a database.

Think about the information that Government should have about each citizen and fill in the blanks. To give you track, some blanks are already filled up.

INFORMATION THAT WILL BE VISIBLE ON THE CARD:

- Name
- Date of Birth
- Photograph
- _____
- _____
- _____
- _____
- _____

INFORMATION THAT WILL BE ENCODED ON THE CHIP:

- Facial image
- Fingerprints
- _____
- _____
- _____
- _____
- _____

INFORMATION THAT WILL BE STORED IN THE DATABASE

- Signature
- biometric information
- _____
- _____
- _____
- _____
- _____

These days some people feel that everyone's DNA numeric profiles should also be available in database. It would really help to combat crime. However, a lot of people are worried about this idea; they think that it would have disadvantages.

Brainstorm and discuss the advantages and disadvantages of storing DNA profiles also in the database.

IT APPLICATIONS

Learning Objectives

After studying this lesson the students will be able to:

- Differentiate between front-end and back-end of an application.
- Identify various components of the front-end of an application.
- Design and develop simple IT applications.
- List the impacts of ICT on society

We have already seen in class XI that IT applications are essential requirement of every individual and organization to simplify their day-to-day work, efficiently manage and execute projects. These applications save time and efforts both. Now, it is the time to get into the real world of IT applications by first exploring the existing applications and then developing new applications to solve real life problems.

In class 11, you have already learnt about broad categories of IT application as e-Gaming, e-Business, e-Governance, e-Learning etc. e-Business involves applications dealing with buying and selling of products and services. e-Governance involves applications which are used by government agencies/organizations to provide better governance. e-Learning involves applications which are developed to help learning of any concept/skill. Similar applications are also possible in other sectors of economy and social service.

You must have used or seen others using many such applications several times. Whenever you perform an activity online, like register for a new email account, apply for a Visa while going abroad, reserve a seat on a flight/train, buy a book online - you are actually using IT applications only. So, you can see how these applications save us time and efforts in getting various jobs done. These applications have become an integral part of our modern society.

In this chapter let us learn about what goes into developing such applications. We shall also take up a few examples for better understanding.

Front-End Interface

All IT applications process some data entered by the user. For example, when an examinee has to see his result on the net, he has to enter his roll number. When a person has to deposit his house tax online, he has to enter information about his house and his credit/debit card using which the house tax has to be deposited. To place an order online for some purchase, the buyer has to enter some information about himself and the item to be purchased. Similarly for any IT application the user has to enter some data which may be just a number or a lot of data like buyer's details. Every IT application provides some sort of form using which users enter the data. This form is called the Front End Interface (or just Front-End or Interface or user-interface) of the application.

To create a front-end various components, like those studied in Java GUI application development, are used. Some of the most commonly used components are discussed below.

TextField: TextField is used to get small textual information like Name, RollNo, email address, quantity, etc. Disabled/Uneditable TextFields are also used to display such information.

TextArea: TextArea is used to get long textual information which may span multiple lines of text. E.g. to get Address, Complaint, Suggestion etc. Disabled/ Uneditable TextAreas are also used to display such information.

Radio Button: Radio buttons are used to get an option out of several mutually exclusive (out of which only one can be selected) options. Examples of such options are Gender (Male or Female or Other), Type of Credit Card (Master or Visa or Other), Type of internet connection (DialUp or Braodband), etc.

CheckBox: Check boxes are used to get one or more options out of several given options which are not mutually exclusive. These are the cases where multiple options are given to the user and the user can select zero or more out of the given options. Examples of such options are Hobbies (a user may have zero or more hobbies), Magazines to subscribe for (a user may subscribe to zero or more of the given magazines) etc.

List: A list is used to get one or more options out of several given options which may or may not be mutually exclusive. This may seem to be the case where CheckBoxes are to be used, but the difference is in the number of options available. If the number of options is small, then CheckBoxes can be used. In case of large number of options, using CheckBoxes may take up a lot of space on the form and it may also be inconvenient for the user to select the desired options. In such cases Lists are preferred over checkboxes. Examples of such cases are: To select cities out of a given list of cities, to select magazines out of a given list of magazines, etc.

ComboBox: A ComboBox is used to get an option out of several given options which are mutually exclusive. This may seem to be the case where RadioButtons are to be used, but the difference is in the number of options available. If the number of options is small, then RadioButtons can be used. In case of large number of options, using RadioButtons may take up a lot of space on the form and it may also be inconvenient for the user to select the desired option. In such cases ComboBoxes are preferred over radio buttons. Examples of such cases are: To select a city out of a given list of cities, to select a train out of a given list of trains, etc.

When the options are mutually exclusive, then a List can also be used instead of a ComboBox. It all depends on the space available on the form (a ComboBox consumes less space as compared to a List) and the look of the form (which the form designer has to decide).

PasswordField: A PasswordField is used to get some secret textual information like Password, CVV number of a credit card etc.

Front-end interface is the face of any application. In most of the cases, the front-end decides whether the application will be easily accepted or not. If the front-end is convenient and unambiguous for the user, then the user will like to use it and hence the application will be given positive reviews. If the front-end interface is inconvenient for the user, then the user will not like to use the application. Therefore, front-end of an

application must be user-friendly. Following are a few tips to make the front-end more and more user friendly:

1. **Consistency:** Consistency in looks and operations plays a major role in front-end design. If in one window the buttons are placed at the bottom, then in all the other windows also they should be placed at the bottom. If double-clicking an item pops-up a short-cut menu, then double-clicking any other item should pop-up the relevant short-cut menu. Labels, color-scheme etc. should also be consistent through-out the application. Consistency enables users to make an idea of how the application works, and this idea leads to fast acceptance of the application.

2. **Make it convenient for the user:**

   a) Place the most important items at the top-left position of the form. When a user looks at a window, top-left is the first position where user attention goes. So an item placed at top-left position has least chances of getting skipped.

   b) Don't use such bright colors which put pressure on users' eyes. The colors which look very fantastic are not necessarily convenient for the user when it comes to entering data or viewing reports.

3. **Help the user enter correct data in the first go:** Ask for minimum textual data to be entered by the user. If you have to ask for class and section, provide a list to choose the class, provide radio buttons to choose the section. This way user has the options only to enter the valid data. If you ask the user to enter the class and section in a text box, then the user has all the options to enter the data and hence more chances of entering invalid data.

4. **Listen to all:** Before creating the user interface, you should speak to the potential users and get their ideas to decide the design of user interface. You should put a limit there only. You must get the ideas but you are not bound to use these ideas. Use your skill and commonsense to decide which of these should be incorporated and which one should not be. The aim is to create a consistent, convenient, and logically correct user interface.

5. **Smooth shifting from one window to the next (or the previous):** Make the sequence of moving from one window to another exactly same as the flow of work the application is made to do.

Back-End Database

Front-End is just one part of an IT application. Any IT application usually stores a lot of data in the form of a database which is not visible to the user. This database is used by the application to give suitable responses to the user. This database is called Back-End Database (or just Back-End or Database). For Example, the database of train reservation system stores all the data about trains and passengers, the database of an online shopping system stores the data of all the items available in the store, and so on. If the front-end interface makes the user like or dislike the application in the first go, then the back-end decides whether the user will keep liking the application or not. A good back-end improves the speed of the application. A good back-end also ensures easy modification of the application whenever required. Following are the features of a good back-end database:

- It should use multiple tables for storing data to avoid data redundancy.
- Tables in the database should be created using constraints wherever applicable.
- Keys (Primary and Foreign) of tables must be defined.

To make the application efficient and effective, you should also follow the guidelines given below:

1. It should meet all the requirements of the problem, for which the application was created.
2. It should have user-friendly interface to make the user comfortable while using.
3. Code should have sufficient number of comments to help the programmer/yourself to modify/update the code in future.
4. Keep the navigation of input in a standard order as much as possible. Most significant information should be entered first.
5. There should not be any ambiguity in data and information and it should avoid inputting duplicate information anywhere in any form.

Front-End and Database Connectivity

A database application consists of Front-End and Database (Back-end). These two entities cannot work in isolation. Whatever data is entered by the user has to go to the database

and whatever relevant data is extracted from the database is to be shown to the user through the Front-End. Therefore, the Front-End and the Database of an IT application must be connected. This connectivity is achieved as learnt in Chapter 6 (Database Connectivity). If the application is web based then the connectivity is achieved using some scripting language (like vbScript or JavaScript).

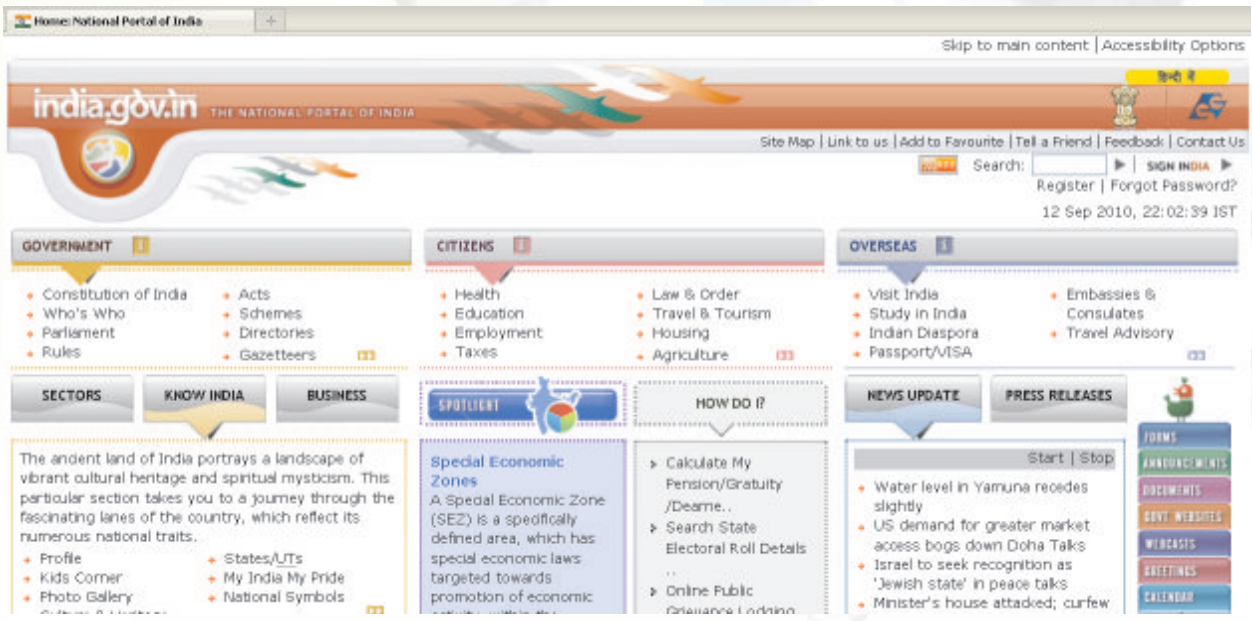
Examples of IT Applications

There are numerous IT applications. We consider herein IT applications for e-Governance, e-Business, and e-Learning. Web addresses of a few of these are given below:

e-Governance:

To reach the citizens in an effective and transparent manner ICT enabled counters have been setup by government where several services like Birth/Death certificate registration, Railway enquiry and ticket booking, submission of RTI application etc. are provided. These centres are accessible to anyone and people can use these to get guidance, information, and services without paying any money to touts or middle men.

1. **india.gov.in (The National Portal of India) -** This portal not only gives the information about Government of India, but also allows the users to apply online for various services provided by the government.



2. **goidirectory.nic.in (Government of India Web Directory) -** Through this portal one can access various government web sites. These sites include sites of various states and union territories, and sites of central government departments etc. All these sites are examples of e-Governance applications of IT. Some of these sites are:

a) **`mcchandigarh.gov.in:`**

Portal of Municipal Corporation of Chandigarh

b) **`Jammukashnir.nic.in:`**

Portal of Municipal Government of Jammu and Kashmir

Know More!

Bhoomi (meaning land) is the project of on-line delivery and management of land records in Karnataka. It provides transparency in land records management with better citizen services and takes discretion away from civil servants at operating levels.

The Revenue Department in Karnataka, with the technical assistance from National Informatics Centre (NIC), Bangalore, has built and operationalised the BHOOMI system throughout the state. The BHOOMI has computerized 20 million records of land ownership of 6.7 million farmers in the state.

Know More!

National Informatics Centre (NIC) is a premiere Science & Technology institution of the Government of India, established in 1976, for providing e-Government / e- Governance Solutions adopting best practices, integrated services and global solutions in Government Sector.

Hindi version of Government of India portal is http://bharat.gov.in/. The image of home page of this website is shown below:

e-Business:

To reach the customers and business associates in an effective and fast manner business houses (now a days many small shops like snacks corners and paan shops also) provide their services on the net. These ICT enabled counters are used to get orders and feedbacks from the customers and also for inter-business transactions. This helps the businesses to widen their customer base.

1. **nafed-india.com/ebusiness.asp (e-business site of NAFED) -** Through this URL NAFD (National Agricultural Cooperative Marketing Federation of India Ltd.) offers its e-business services to various corporates and customers.

2. **Amazon.com (e-Business site of Amazon.com) -** Amazon is the world's largest online store. Through this URL Amazon does its online business

e-Learning :

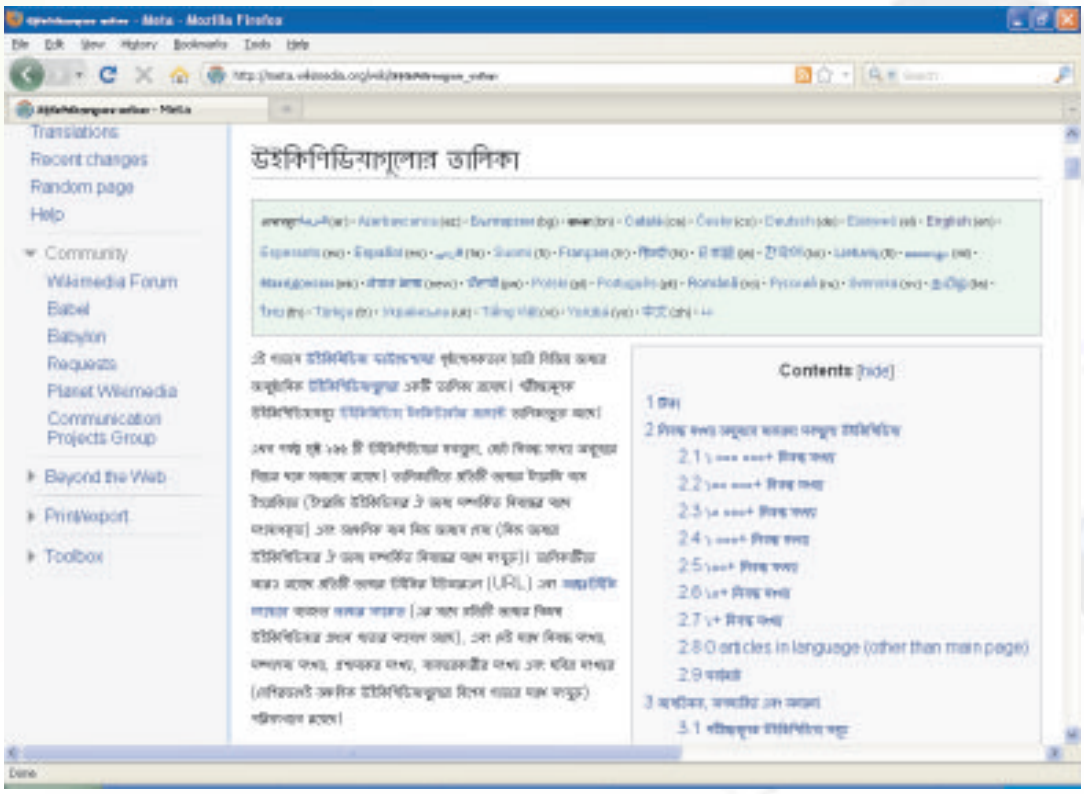
e-Learning has multiple goals. It is much more than having a net connection and/or CDs through which people learn. E-Learning is about giving freedom to people to learn whatever they want to learn and whenever they want to learn. This is irrespective of (except in exceptional cases) age, caste, gender, economical background, or qualification of the learner. The only requirement is the will to learn. E-learning is available on almost all the topics imaginable.

1. **w3schools.com (Website Developers e-Learning site) -** At w3schools.com you will learn how to make a website. It offers free tutorials in all web development technologies.
2. **www.gcflearnfree.org -** It is an educational part of the GCF mission. GCF creates and provides quality, innovative online learning opportunities to anyone who wants to improve the technology, literacy, and math skills necessary for them to be successful in both work and life. GCF believes that there's freedom in the ability to learn what you want, when you want, regardless of your circumstances.
3. **educationportal.mp.gov.in/public/multimedia.aspx -** This government of Madhya Pradesh portal provides multimedia tutorials on various topics of different subjects like maths, science, social sciences etc.
4. **ncert.nic.in/html/learning_basket.htm -** This NCERT portal provides interactive modules for students to learn various topics.

Websites in Indian languages

These days multiple Government and private organizations are providing their websites in Hindi and other regional languages also. The aim is to provide their services even to the common people in remote areas. Small towns where computers and internet have reached, information on the net should also be available in regional languages so that people not knowing English can also have access to the information. Language should not be a hinderance but a support to learning. Understaning the importance of regional languages, many websites have also provided translation services so that the same page can be viewed in any language of user's choice. Following are the home pages of a few websites in Hindi and other regional languages:


www.invata-mate.info/indusa/index.htm
This page is in Hindi Would you like to translate it? Translate Nope Options
math.info
FLASHGAMES247.COM
3D SUDOKU


Translations
Recent changes
Random page
Help
Community
Wikimedia Forum
Babel
Babylon
Requests
Planet Wikimedia
Communication Projects Group
Beyond the Web
Print/export
Toolbox
উইকিপিডিয়াগুলোর তালিকা
Contents [hide]


Google
ਅਮ੍ਰਿਤਸਰ
ਤੇ ਲੱਭੋ
ਵੈੱਬ
ਅਮ੍ਰਿਤਸਰ ਲਈ 41,200 ਵਿੱਚੋਂ 1 - 10 ਨਤੀਜੇ (0.24 ਸੈਕਿੰਡ)
ਹੁਕਮਨਾਮਾ ਦਰਬਾਰ ਸਾਹਿਬ ਸ੍ਰੀ ਅਮ੍ਰਿਤਸਰ - DC Forum
ਅਮ੍ਰਿਤਸਰ: ਬਲੂਸਟਾਰ ਦੇ ਵਿਰੋਧ ਵਿਚ ...
ਖੂਨ ਚ ਡੁੱਬ ਡੁੱਬ ਰੋਇ, ਧਰਤਿ ਅਮ੍ਰਿਤਸਰ ਦੀ - Saanj
ਅਮ੍ਰਿਤਸਰ 'ਚ ਭਾਰੀ ਮੀਂਹ - Punjabi News


ताज महल
प्रस्तावना
इतिहास
वास्तु कला

Know More!

Guidelines for Multilingual websites:

NIC has developed guidelines for Indian Government websites. These guidelines are accessible at http://www.pon.nic.in/homeinfo/govt-website-guidelines.pdf. Article 5.7 of this document lays guidelines for Multilingual versions of Government websites. The main points of this article are:

a) Ideally all the pages on the website should be translated in Hindi and other regional languages. In case it becomes difficult to do so, corresponding Departments should identify the content which is widely accessed by the public and put up such content in regional languages.

b) It MUST be ensured that the documents/pages in multiple languages are updated simultaneously so that there are no inconsistencies, at any point, between the various language versions.

c) In case it is practically difficult to update the versions in all the languages simultaneously due to delays on account of translation etc., the obsolete information should be removed from the site till the latest information is uploaded. In any case, a time stamp indicating the date of uploading the information and its validity should be put along with all the time sensitive documents.

Development of IT applications

After going through these various IT Application sites, you must have realied that all these applications simplify the processes. Now, let us find out how these applications are created.

For developing such application, one needs to follow the following steps:

Step 1: Identify the problem for which the application is to be developed and discuss about its feasibility. If the applications is technically and economically feasible (possible and profitable to carry out), then steps are taken for its development, otherwise the project is scrapped.

Step 2: Identify and decide, which database tables and table structures will be required in the application. Make sure that the data types and sizes of the columns in the tables are carefully planned and used. Create database and tables as per the requirement of the application.

Step 3: Identify and decide, which all inputs are required to be taken from the user in the Front-End of the application. Find out, where you can minimize the typing efforts of user by introducing known options using RadioButton/CheckBox/ List/ComboBox etc. Develop the front-end of the application as per the requirement and ease of use.

Step 4: Establish the data connectivity between the Front-End interface and Back-End Database.

Step 5: Test the full application (Front-End and Back-End) with multiple sample sets of data. It is always better if the sample data are collected from potential users of the application randomly.

Now, the application is ready for implementation.

Examples: We shall take 3 examples, one from each of the categories: e-Business, e-Governance, and e-Learning, to better understand the process of IT application development.

Example 1 - e-Business :

Let us see how can we develop the application which we described as 2nd Team based exercise in chapter 9. To recall it, it is reproduced below:

To expand its business, XYZ Mall plans to go online. Anyone who shops at the Mall will be given a membership number and Password which can be used for online shopping. With this membership number and password, customers can place their orders online. The mall will maintain the customers' data and orders' data. A person is put on duty to keep constantly checking the Orders data. Whenever an order is received, its processing has to start at the earliest possible.

The Orders' data will be analysed periodically (monthly, quarterly, annually - whatever is suitable) to further improve business and customer satisfaction.

The application development steps are given below:

Step 1: The problem is identified. Its economic and technical feasibility are discussed. It is found that the application development and implementation are technically and economically feasible. So the management gives a green signal for application development.

Step 2: After discussing with the management and the potential users of the application, it is decided to create the following tables in the database:

Table: Customers

S.No.	Field	Datatype	Remarks
1.	Membership_number	Int(9)	Primary key. First 6 digits will identify the membership date and the last 3 digits the serial number on that date.
2.	Password	Varchar(15)	-
3.	Name	Varchar(25)	Customer's Name
4.	Address	Varchar(50)	Customer's address.
5.	Phone	Varchar(30)	Customer's phone number(s).

Table: Items

S.No.	Field	Datatype	Remarks
1.	Item_Code	Int(5)	Primary key. First 2 digits will identify the category of the item, and the last 3 digits the serial number in that category.
2.	Description	Varchar(30)	Item's Name
3.	Price	Decimal(7,2)	Item's Price
4.	Discount	Decimal(5,2)	Percentage Discount allowed on the item.

Table: Orders

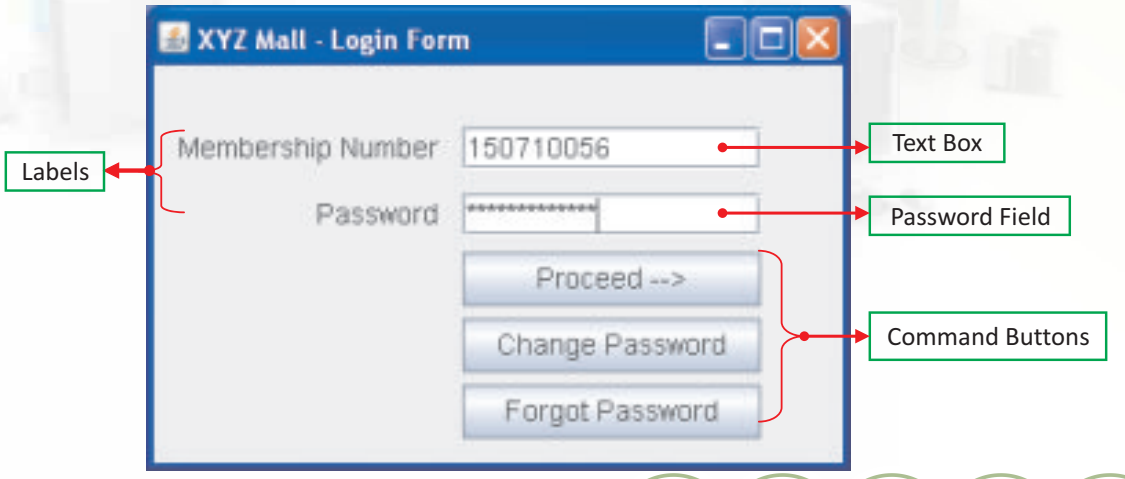
S.No.	Field	Datatype	Remarks
1.	Ord_number	Int(5)	Order Number.
2.	membership_number	Int(9)	Membership number of the customer who has placed this order.
3.	Ord_date	Date	Date of order placement.
4.	Item_Code	Int(5)	Code of the item ordered.
5.	Qty	Decimal(5,1)	Quantity of the item ordered.
6.	Status	Varchar(10)	Current status of this item. Status can be: Waiting/Completed/Cancelled.

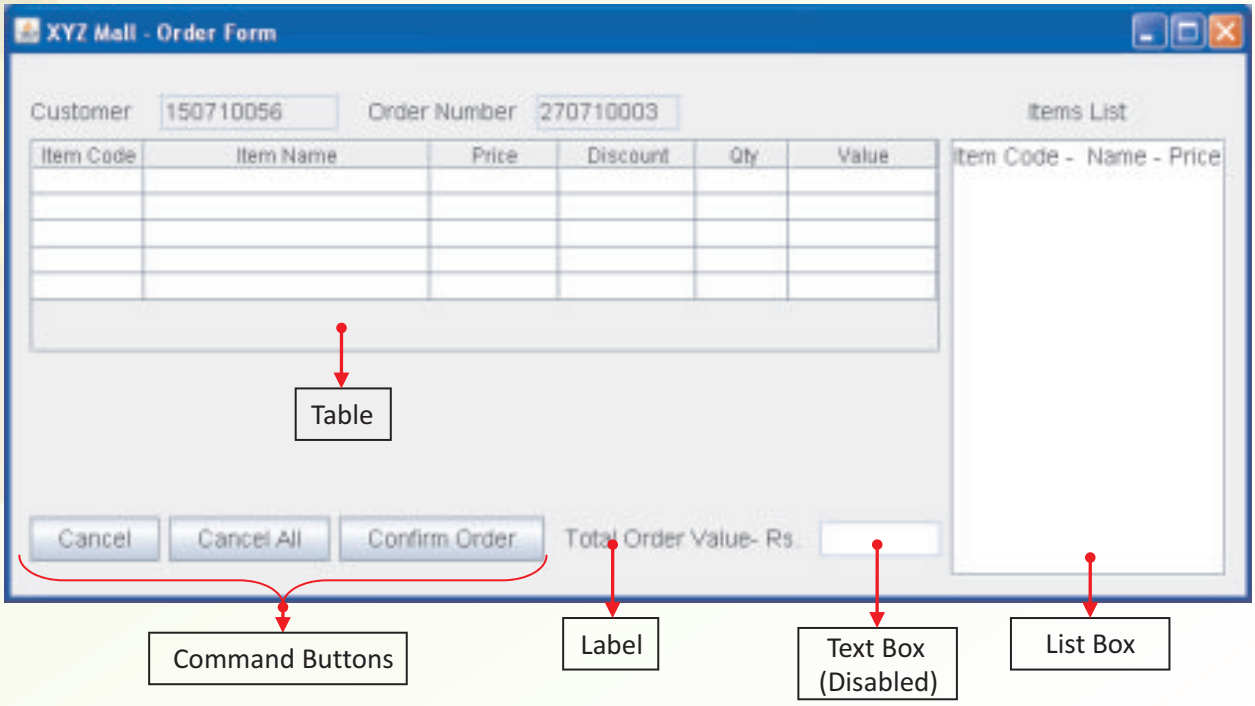
Step 3: After careful analysis, it is found that the inputs required from the user are as follows:

- Membership_number
- Password
- Item_codes of the items to be ordered
- Quantities of the items ordered

All the other fields like order_number, item_name, Order_date, Discount etc. will be auto-generated/calculated by the application.

To get input from customers, following front-end interface can be designed:





Step 4: Data connectivity between the Front-end interface and back-end database is established using some scripting language like Javascript or VBScript etc.

Step 5: After development the application is tested by giving membership numbers and passwords to the employees of the mall itself so that the application can be tested with multiple sets of random data.

If any discrepancies are found during test, the corresponding corrections are made in the application. After complete testing, the application is implemented and the customers are given membership numbers and passwords.

Example 2 - e-Governance :

The state administration wants to make vehicles' data (RegNo, RegDate, Owner, OwnerShipNumber, Address, HP) easily available to citizens. Each registration authority in-charge will regularly update the data. Citizens will be given *read only* access to this data.

The application development steps are given below:

Step 1: The problem is identified. Its economic and technical feasibility are discussed. It is found that the application development and implementation are technically and economically feasible. So the corresponding authority (may be the State Transport Authority) gives a green signal for application development.

Step 2: After discussion with the management and the potential users of the application, it is decided to keep the following tables in the database:

Table: Vehicle

S.No.	Field	Datatype	Remarks
1.	RegNo	Char(12)	Primary key. Resgistration Number is 10 characters long. To make it future proof, a provision of 12 characters is made.
2.	RegDate	Date	Registration date of the vehicle.
3.	VehicleType	Varchar(10)	Car/Truck/Bus/Scooter/MCycle etc.
4.	Manufacturer	Varchar(20)	Vehicle Manufacturer's Name
5.	Model	Varchar(20)	Name of the Model

Table: Owner

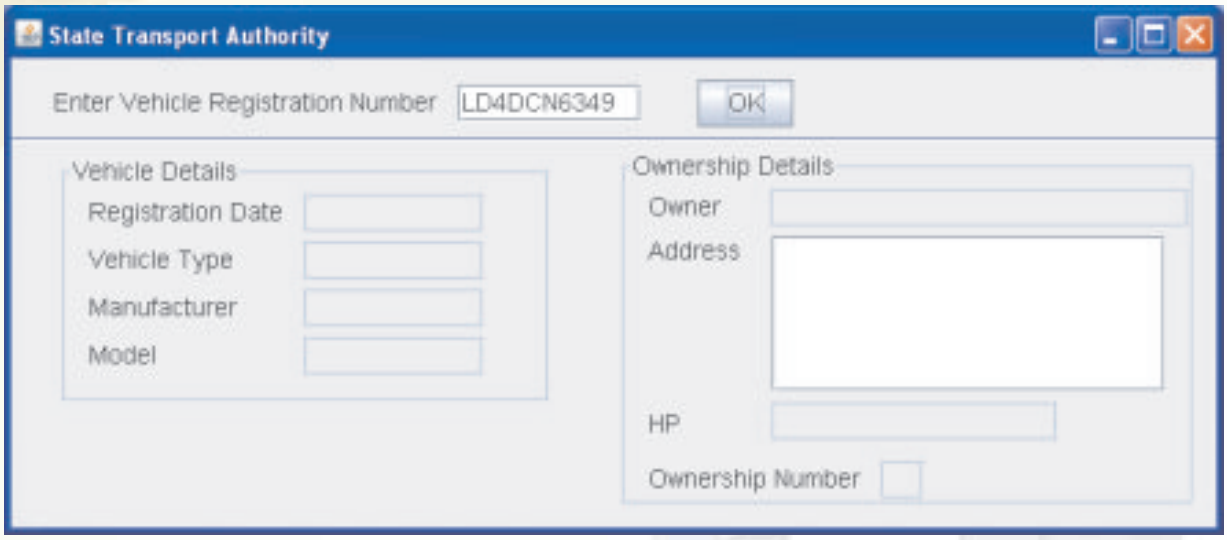
S.No.	Field	Datatype	Remarks
1.	RegNo	Char(12)	Primary Key.
3.	Owner	Varchar(25)	Owner's Name
4.	Address	Varchar(50)	Owner's address.
5.	OwnerShipNo	Int(2)	Ownership Number of the vehicle.
6.	HP	Varchar(25)	Name of the bank/institution which has financed the vehicle.

Step 3: After careful analysis, it is found that the inputs required from the user are as follows:

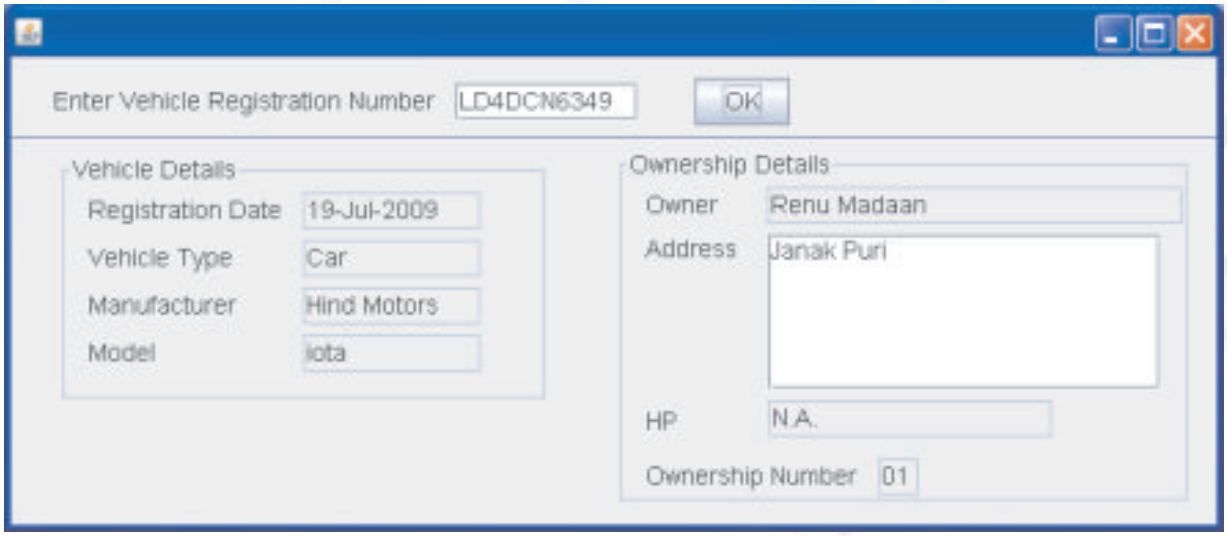
- Registration Number

All the other fields like Owner's Name and Address, HP etc. will be provided by the application from the database.

To get input from the users, following front-end interface can be designed:



Once the registration number is input, database can be searched for it and the corresponding details of the vehicle and the owner can be shown to the user in the following format:

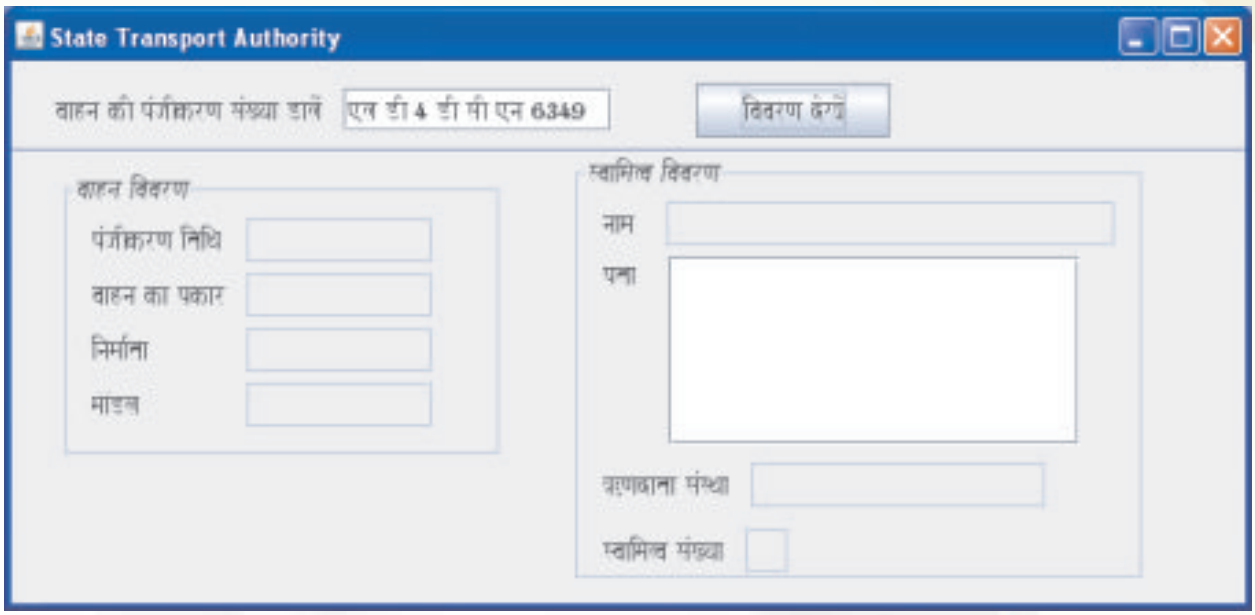


Step 4: Data connectivity between the Front-end interface and back-end database can be established using some scripting language like Javascript or VBScript etc.

Step 5: After development the application can be tested by entering various registration numbers.

If any discrepancies are found during test, the corresponding corrections can be made in the application. After complete testing, the application can be implemented.

Hindi version of this IT application can look like as follows:



Example 3 - e-Learning :

An organization of dedicated teachers, 'Meticulous Teachers Consortium', decides to invite computer aided teaching modules from individuals and organizations so that these can be put on the internet for students' use free of cost. No money will be charged from users and no money will be paid to the developers. Once the modules start pouring in, a front-end is created for the students where the students can select any of the available modules to learn any topic.

The application development steps are given below:

Step 1: The aim is identified. Its economic and technical feasibility are discussed. It is found that the application development and implementation are technically and economically feasible. So it is decided to go ahead with the application development.

Step 2: To keep track of various modules contributed and used, the following database is created:

Table: Modules

S.No.	Field	Datatype	Remarks
1.	ModuleNo	Int(5)	Primary key. Serial Number allotted to the module.
2.	Subject	Varchar(20)	Subject name to which the module caters.
3.	Topic	Varchar(20)	Topic Name to which the module caters.
4.	Level	Varchar(10)	Beginner/Intermediate/Advanced.
5.	Duration	Int(4)	Duration of the module in seconds.
6.	Developer	Varchar(20)	Name of the module developer.
7.	SubmissionDate	Date	Date on which the module is submitted.
8.	NoOfHits	Int	Number of times the module is viewed by the users. Each time a user views the module, NoOfHits is incremented by 1.
9.	LastUsedDate	Date	Date on which the module was viewed most recently.

Step 3: After careful analysis, it is found that the inputs required from the user are as follows:

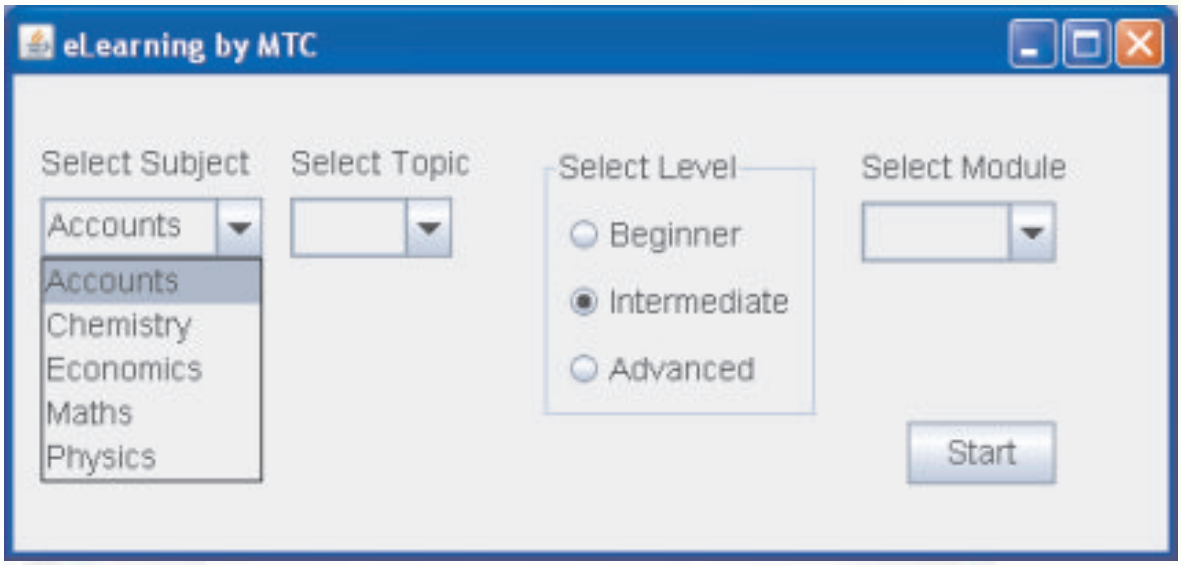
If a user has already viewed a module and he again wants to view the same module which he remembers by its module number, then the only input required is:

- Module Number

If a user wants to see a module for the first time or he does not remember the module number, then the input required is as follows:

- Subject (from a list of subjects)
- Topic (from a list of topics corresponding to the subject chosen)
- Level (Beginner / Intermediate / Advanced)
- Module Number (from a list of module numbers corresponding to the above 3 enteries)

To get input from a user, following front-end interface can be designed:



Once the desired input is obtained the corresponding module is executed.

Step 4: Database connectivity between the Front-end interface and back-end database can be established using some scripting language like Javascript or VBScript etc.

Step 5: After development the application can be tested by selecting various modules and running them.

If any discrepancies are found during test, the corresponding corrections can be made in the application. After complete testing, the application can be implemented.

Impact of ICT on society

Like everything else that is used by common man, ICT (Information and Communication Technology) also has impacted the society. ICT has impacted the society in a much wider way than any other technology. Most of these impacts are positive, though there are some negative impacts also.

Social and Economic benefits of ICT:

- Social networking sites help people remain in touch with their nears and dears even when they are staying on opposite sides of the globe.
- Social networking sites help like minded people come together and work for some cause.
- e-Governance sites help people save their productive time by performing various government related jobs like getting some forms, depositing bills online.
- ICT helps economy grow at a faster rate as it provides transparency in the processes and helps the government to keep check on defaulters.
- Due to e-Banking and use of plastic money more money is put in circulation leading to faster growth of GDP.
- e-Learning sites make quality study material available even to the students staying at remote places.

Infomania:

Infomania is the condition of reduced concentration caused by continually responding to electronic communications such as e-mail, SMSs, MMSs etc. ICT is making more and more people infomaniac. This is making some people waste their productive time in the office, neglect their families and duties. Some people are also in a habit of frequently checking their e-mails even when they are on vacation with their families. We have to be careful in the use of ICT so that we use it constructively and not get obsessed with it and become infomaniacs.

Summary

- Three major groups of IT applications covered in this chapter are: **e-governance, e-business, and e-learning.**
- **e-Governance** involves applications which are used by government agencies/organizations to provide better governance.
- **e-Business** applications use technology to effectively access and deliver business related services and perform various kinds of business transactions.
- **e-Learning** applications use technology to effectively deliver and monitor learning and teaching processes. They help the trainer to organize and manage his/her lesson plans, present them to students/learners, evaluate and take the feedback to enhance & fine-tune this process in future.
- An IT application has two major parts: Front-end (The user interface) and **back-end** (The database)
- The **front-end** of an IT application is usually a group of one or more forms through which the user enters the input values and is shown the corresponding output. A good front-end ensures the acceptance of the application in the first go.
- The **back-end** of an IT application is the database in which all the data is stored. This database resides in the server. All the data which is requested by the front-end is supplied by back-end. A good back-end ensures sustainability, efficiency and easy modification of the application.
- Development of an IT application involves creation of front-end, back-end, and connecting these two. It also involves testing the application and then implementing it.
- Use of ICT has its social and economic impacts. Society is impacted as due to ICT people change their way of conducting the transactions and thus save their time, money, and energy. Economy is impacted as ICT leads to fast completion of data transfer and data processing jobs. ICT also brings transparency in the administration.

EXERCISES

MULTIPLE CHOICE QUESTIONS

1. A web site to provide online information and services to the citizens is an example of

   a. e-Business b. e-Mail

   c. e-Governance d. e-Learning

2. The web-site of an electricity supply company which allows its customers to pay bills online is an example of

   a. e-Business b. e-Mail

   c. e-Governance d. e-Learning

3. The web-site of a school which allows the students to go through various lessons in their subjects is an example of

   a. e-Business b. e-Mail

   c. e-Governance d. e-Learning

4. Web address of national portal of India is:

   a. India.gov.in b. GOI.gov.in

   c. ncert.nic.in d. None of the above

5. A form through which users interact with an IT application is a part of

   a. database b. front-end

   c. back-end d. Javascript

6. A good front-end is

   a. consistent b. user-friendly

   c. neither of the above d. both a and b.

7. Mr. X is an infomaniac. It means he

   a. Uses information carefully

   b. Uses computers to get information.

c. Responds to almost all his SMSs, eMails etc.

d. Tries to get correct information

8. Javascript is a

a. database

b. front end

c. back-end

d. scripting language

ANSWER THE FOLLOWING QUESTIONS

1. What is the advantage of using IT applications over the manual operations?
2. Write two important features of each of the following types of applications?
   a. e-Governance
   b. e-Business
   c. e-Learning
3. Give some examples of input values, where Radio Button and Check Boxes should be used for efficiency in the application.
4. What are the important guidelines we should keep in mind while developing an efficient application?
5. Is it a good practice to take in the inputs using TextFields only? Justify your answer.

TEAM BASED TIME BOUND EXERCISE:

(Team size recommended: 3 students each team)

1. **Online Applications for New Water Connection:** Municiple Corporation of Indirapuram has decided to develop an IT application so that citizens can apply online for new water connections. A rough idea of how the application will function is given below:

   Application forms for new water connection will be made available online. These forms can be downloaded from the net, filled in, and submitted online only.

   When an application is received, the server automatically puts it in the table of corresponding zone. Zonal files are accessible by respective zonal heads.

It is responsibility of the zonal head to ensure that an application from his/her zone is processed within a stipulated time period. Whenever a new connection is granted or rejected, the ResponseDate, and Remarks columns of corresponding row are updated by the respective zonal head. For this purpose, one of the tables of the database is given below:

S.No.	Field	Datatype	Remarks
1.	App_Number	Int(9)	Primary key. This is the application number of the received application. App_number is auto-generated by the application. First 6 digits will identify the date of application and the last 3 digits, the serial number of the application received on that date.
2.	NameOfApplicant	Varchar(30)	Name of the applicant
3.	Address	Varchar(50)	Address of the applicant
4.	AppDate	Date	Date of application
5.	Conn_Type	Varchar(10)	Type of connection required. It can be domestic/commercial/charity.
6.	Zone	Varchar(10)	Zone of the applicant's residence. It can be East, West, South, North, Centre.
7.	ResponseDate	Date	Date when the action on the application is started by the authorities.
8.	Remarks	Varchar(30)	Remarks to be entered by the corresponding zonal head. It can be In Progress / Rejected / Completed /Incorrect Address.

The job of each team is to describe the remaining tables of the database and to suggest the design of front-end.

2. **Birth Certificate:** Babu Lal lives in a small village. He is blessed with a girl at home only. The village teacher visits his house and congratulates him. She tells him to register the birth of his daughter with the government, and shares the benefits of having the Birth Certificate. Birth certificate will help Babu Lal's daughter name inclusion in ration card, at the time of admission to school, to avail facilities like Ladli etc. The information base is used for forecasting and planning activities related to healthcare, Welfare measures etc. in an effective way. Babu Lal is happy and immediately goes to the centre for his daughter's Birth certificate generation.

   Your job is to develop an application which generates birth certificates. The database for this purpose is as follows:

S.No.	Field	Datatype	Remarks
1.	BirthID	Int	This is th primary key of the table.
2.	Gender	Char(1)	'M', 'F', or 'O'
3.	DateOfBirth	Date	
4.	TimeOfBirth	Varchar(10)	Time of birth in hh:mm:ss format according to 24hr clock.
5.	PlaceOfBirth	Varchar(30)	
6.	FatherName	Varchar(30)	
7.	MotherName	Varchar(30)	
8.	ResAddress	Varchar(30)	
9.	IssuingAuthority	Varchar(30)	

The birth certificate can be of the form:

Name / नाम : _____

Gender / लिंग : _____

Date of Birth / जन्म की तारीख : _____

Time of Birth / जन्म का समय : _____

Place of Birth/ जन्म स्थान : _____

Hospital Name/ अस्पताल का नाम : _____

Delivery Type / प्रसूति का प्रकार : _____

Father's Name / पिता का नाम : _____

Mother's Name / माता का नाम : _____

Residential Address / घर का पता : _____

Issuing Authority / जारी करने वाला अधिकारी : _____

3. **Sales Agency (B2B):** A sales agency gets FMCG from manufacturers and sells these to Whole sellers. For this the agency has a Suppliers table, a Goods table, and a Customers table.

   Customers can place their orders online. The details of orders accepted are placed in an Orders table.

   The Goods table is regularly updated. Orders table is checked twice a day for any new orders.

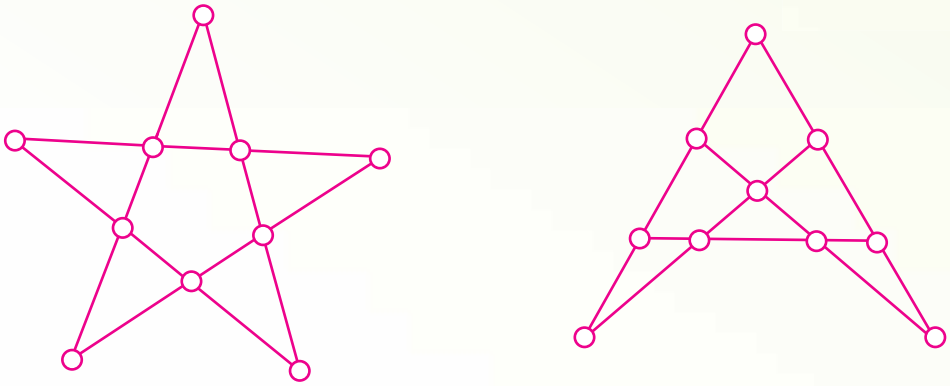
   The job of each team is to describe the attributes with data-types to be placed in each table and and to suggest the design of front-end.

APPENDIX - I

The teacher should divide the class in teams of 3-4 students each and challenge teams to solve the given puzzle. The teams that solve the puzzle within 3 minutes, 4 minutes, and 5 minutes are awarded the title of 'Fasttrack', 'Achiever', and 'Able' respectively.

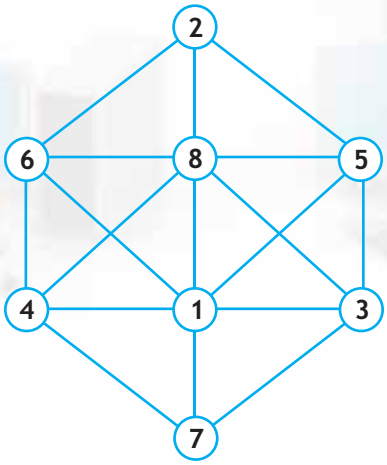
Solution 1

There are two different ways to place ten coins satisfying the given conditions.



Solution 2

We are giving you one possible solution but there are many other solutions possible. So keep trying.....



Solution3

A List Of Palindromes (Phrases). Each line of the phrase is a palindrome.

Solution 4

The lady's children are aged 2, 2 and 9.

As per the first two conditions that she has 3 children and the product of their ages is 36(6 runs each in 6 balls = 6 x 6) the probable ages are:

(1,1,36),(1,2,18),(1,3,12),(1,4,9),(1,6,6),(2,3,6),(3,3,4) and (2,2,9).

The student knows her berth number but he is not able to reach the answer. That means the sum must be same for two of the pairs. This reduces the chances to two: { (2,2,9),(1,6,6) }.

From the third option, we get a constraint that, there is only one elder child. And we reach the correct answer of (2,2,9)

Solution 5

9-digit number is: 473816952

First rounding (down) yields: 473816950	(as 2 rounds down to zero, 5 stays the same)
2nd rounding (up) yields : 473817000	(5 rounds up to zero, add the carried over one to 9 to get 0, add the carried over one to 6 to get 7)
3rd rounding (down) yields: 473817000	(0 rounds down to zero, 7 stays a seven)
4th rounding (up): 473820000	(7 rounds up to zero, 1 adds one and becomes 2)

At this point sum of all the not rounded digits is 2+8+3+7+4 =24

5th rounding (down) yields: 473800000	(2 rounds down to zero, 8 stays a eight)
6th rounding (up) yields: 47400000	(8 rounds up to 0, 3 adds one and becomes 4)
7th rounding (down) yields: 47000000	(4 rounds down to 0, 7 stays a seven)
8th rounding (up) yields: 50000000	(7 rounds up to 0, 4 adds one and becomes 5)

Solution 6

One paise = 1/100 of a rupee.

Given the conditions, we know that:

(100*X + Y)*2 = Y*100 + X - 20

200X + 2Y = 100Y + X - 20

199X - 98Y = -20

We also know that X and Y must both be integers.

So, putting the equation in Y = MX + B form:

199X + 20 = 98Y

(199X + 20)/98 = Y

Now put this in the Y= function of a graphing calculator, set the table to start at 1 and go up by increments of 1 and then scroll down the table until you see an integer pair.

The integer pair occurs at (26,53)

Therefore, X = 26 and Y = 53

Solution 7

1. Mark the jars with numbers 1, 2, 3, 4, and 5.
2. Take 1 pill from jar 1, take 2 pills from jar 2, take 3 pills from jar 3, take 4 pills from jar 4 and take 5 pills from jar 5.
3. Put all of them on the scale at once and take the measurement.
4. Now, subtract the measurement from 150 (1*10 + 2*10 + 3*10 + 4*10 + 5*10)
5. The result will give you the jar number which has contaminated pill.

Solution 8

The numbers of the houses on each side will add up alike if the number of houses be 1 and there are no other houses except David's house, or if David's house number be 6 with 8 houses in all, or if David's house number be 35 with 49 houses, or if David's house number be 204 with 288 houses, and so on. But we know that there are more than 30 and less than 50 houses, so we are limited to a single case.

Mr. David's house number must be 35.

Solution 9

Required : 50m + 100n = 2500 ---------- 1

By mistake, Vijaya was given : 100m + 50 n

50m + 100n - (100m+50n) = 500

50m + 100n - 100m -50n = 500

50n - 50m = 500 ---------- 2

Adding 1 and 2

150n = 3000

n = 20 ---------- 3

Substituting value of n in Equation 2.

50 x 20 -50 m = 500

1000-50m = 500

-50m = -500

m = 10

Therefore Vijaya asked for 10 notes of Rs.50.00 and 20 notes of Rs. 100.00

APPENDIX - II

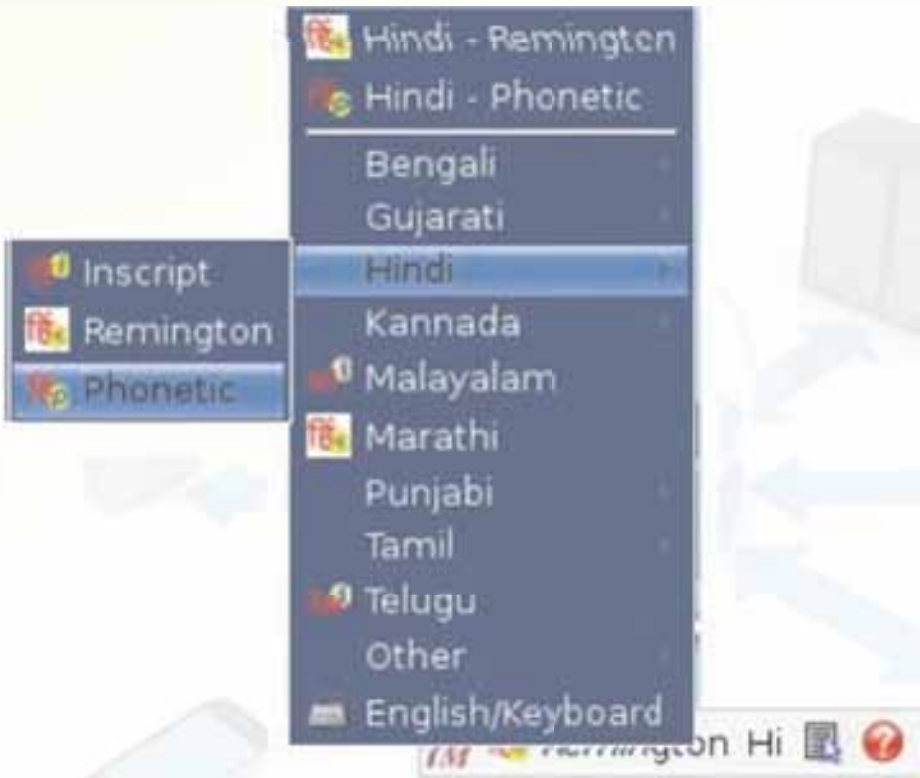
Indic Language Support

Mac OS 10.5 supports Devanagari, Gujarati, Gurmukhi and Tamil

Linux based desktops support Bengali, Devnagari, Gujarati, Kannada, Malayalam, Oriya, Tamil, Telugu and Gurmukhi,

Creating Documents in Multiple Languages in BOSS, GNU/Linux

There are various Input Methods available in BOSS, GNU/Linux. In these operating systems you can select the language of your choice in which you want to create your document. For example to choose Hindi:



Within each language there are multiple keyboard layout options are available. You can choose keyboard layout of your choice.

- Three different type of keyboards are supported
  - i) **Phonetic Keyboard:** प्रक्रिया is typed as 'p r a k r i y a' through English phonetic keyboard.
  - ii) **INSCRIPT Keyboard :** Keyboards are used which contains Indian alphabets as the key of this keyboard. So by typing those keys the content of that language can be written. (REF: www.tdil.mit.gov.in/isciichart.pdf)
  - iii) **Remington Keyboard:** Keyboard also contains keys of the Indian languages and the arrangement of the keys follows the arrangement of a typewriter.

Enabling Indic Language Support in Windows

Windows 7 and Windows Vista include all the necessary files to support Indic languages i.e. Complex(Indic) text support is automatically enabled. Therefore you just need to enable the keyboard for the language that you want to use by following the steps in the Enable a keyboard layout section.

For Windows XP, some additional setup may be required to support Indic languages. Therefore you first follow the steps given under Enabling International Language Support in Windows and then proceed with the steps given under the Enable a keyboard layout section.

Enabling International Language Support in Windows :

Step 1: Click **Start** and then go to **Control Panel.**

Step 2: Click on **Date, Time, Language, and Regional Options** and choose **Add Other Languages** from the task list.

Step 3: In the **Regional and Language Options** dialog box Highlight the **Languages** tab.

Step 4: In the **Regional and Language Options** dialog box, under **Supplemental Language Support,** select the **Install files for complex script and right-to-left languages** check box. Click OK or Apply.

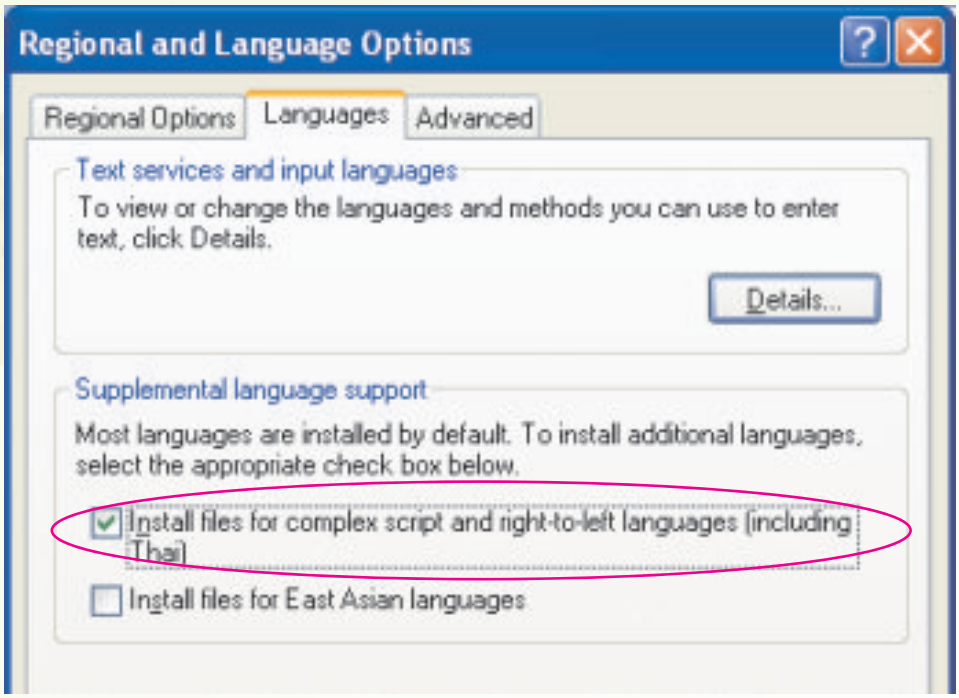


Figure 1 Languages tab in Regional and languages option in Windows XP

Step 5: You will be prompted to insert the Windows CD-ROM or point to a network location where the files are located. After the files are installed, you must restart your computer.

Enable a Keyboard Layout in Windows :

Step 1: Under "Text services and input languages," click on the "Details..." button.

Step 2: Under Installed Services, click "Add..."

Step 3: In the **Text Services and Input Languages** dialog box, on the **Settings** tab, click **Add.**

Step 4: In the **Add Input Language** dialog box, click the **Input language** list and select your preferred language and dialect. If you want to change the standard keyboard layout, click the **Keyboard layout/IME** list and select a new keyboard layout. Then, click **OK.**

Step 5: In the **Text Services and Input Languages** dialog box, on the **Settings** tab, click the **Default input language** list, and select the language you will use most often. The language you select as the default will display when you first start your computer. If you have finished adding languages, click **OK.**

Step 6: Click the **Regional Options** tab. Click the **Standards and formats** list, and then select your region.

Step 7: Click the **Location list,** and then select your location.

Step 8: Once done, click OK to exit. On the Text Services and Input Languages page, click OK again to close Regional Options. You should now see a language indicator in the System Tray (located at bottom right hand corner of the desktop by default)

Turning on the language bar :

If you do not see the language bar in the task bar (at the bottom of the desktop) or floating on the desktop please do the following:

Step 1: Click Start, click Control Panel, and then double-click Regional and Language Options.

Step 2: On the Languages tab, under Text services and input languages, click Details as shown in Figure 8.

Step 3: Under Preferences, click Language Bar.

Figure 2 Turning on the Language Bar in Windows XP

Step 4: Select the Show the Language bar on the desktop check box.

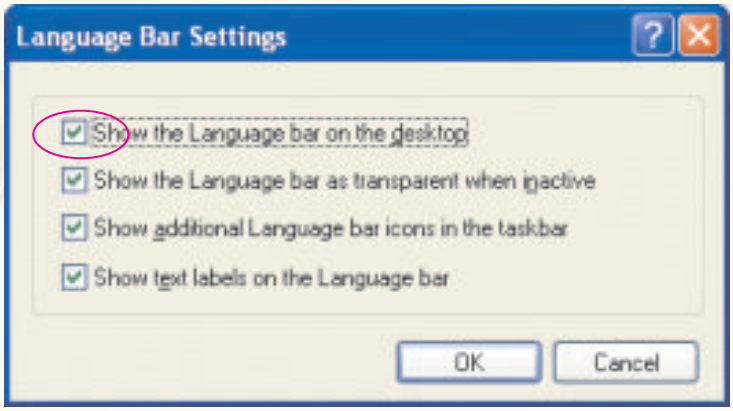


Figure 3 Language Bar Settings in Windows XP

Note: You can switch between different languages by clicking on the language bar and changing the language or by pressing the left ALT+SHIFT keys.

Using Microsoft Indic Language Input Tool ("ILIT") :

Do the following to use Microsoft ILIT in any application, such as Notepad.

Step 1: Open the application in which you would like to enter Indic text.

Step 2: Change the language using the language bar, which typically appears in the taskbar as follows.

Figure 4 - Language Bar in System Tray in Windows XP

Step 3: The language bar will now show the current language.

Step 4: You can now start typing in English and whatever you type automatically gets transliterated after a word-breaking character like a space, comma, etc. is entered. Note that this language setting is per application. You may have to repeat the steps above for each application you want to use Microsoft ILIT in. The Microsoft indic input tool comes with a virtual keyboard that can be used to type in notepad.

Figure 5 Virtual Hindi Keyboard

Note: You can switch between different languages by clicking on the language bar and changing the language or by pressing the left ALT+SHIFT keys.

Installing Fonts in Windows :

Step 1: Go to Windows Fonts folder e.g. C:\Windows\Fonts. (The path may differ on some computers.)

Step 2: Copy-paste the font file into this folder. Windows will now install the font file.

Step 3: Once installed the font will be available in your text-based applications.

Source:

1. http://www.microsoft.com/windowsxp/using/setup/winxp/yourlanguage.mspx

2. http://office.microsoft.com/en-us/support/enable-keyboard-layouts-for-different-languages-HA010354267.aspx