RDBMS concepts and ORACLE

Lab 1

Create the student table using the given fields.

(1) Student Information Table:

```
Student_id          varchar2(20)
Last_name           varchar2(25)
First_name          varchar2(20)
Dob                 varchar2(20)
Address             varchar2(300)
City                varchar2(20)
State               varchar2(2)
ZipCode             varchar2(9)
Telephone           varchar2(10)
Fax                 varchar2(10)
Email               varchar2(100)
```

(2) Department Information Table:

```
Department_Id       varchar2(20) primarykey
Department_Name     varchar2(25)
```

(3) Instructor's Information Table:

```
Instructor_id       varchar2(20) primarykey
Department_Id       varchar2(20) Foreignkey
        department(department_id).
Last_Name           varchar2(25)
First_Name          varchar2(200)
Telephone           varchar2(20)
Fax                 varchar2(20)
Email               varchar2(100)
```

(4) Course Information Table:

```
Course_Id           varchar2(5)
Department_Id       varchar2(20) foreignkey  department(department_id)
Title               char(60)
Description         varchar2(200)
Additional_fees     number primarykey (course_id, department_id)
```

(5) Schedule Type Header Table:

```
Schedule_Id          varchar2(20)
Schedule_Description varchar2(200)
```

(6) Schedule Type Details:

```
Schedule_Id          varchar2(20)
Day                  number
Starting_Time        date
Duration             number
```

(7) Class Location Information:

```
Class_Building       varchar2(25)
Class_Room           varchar2(25)
Seating_Capacity     varchar2(2)
```

(8) Class Table:

```
Class_Id             varchar2(20) primarykey
Schedule_Id          varchar2(20)
Class_Building       varchar2(25)
Class_Room           varchar2(25)
Course_Id            varchar2(5)
Department_Id        varchar2(20) foreign key
                     Dept_info(Department_id)
Instructor_Id        varchar2(20) Instructor(Instructor_id)
Semester             varchar2(6)
School_Year          date
```

(9) Student Grade Information Table:

```
Student_Id           varchar2(20)
Class_Id             varchar2(20)
Grade                varchar2(2)Check
                     Grade in ('A','A+','A-','B','B+','B-',
                     'C','C+',C-','D','D+','D-','F','F+','F-')
Date_Grade_Assigned  date
```

(10) Describe the structure of the following tables.

1. Student information table
2. Department information table
3. Instructor's Information Table
4. Course Information Table
5. Schedule Type Header Table
6. Schedule Type Details
7. Class Location Information
8. Class table

9. Student Grade Information Table.

Lab 2

Alter the Table with the following requirements.
(Hint: Use Alter Table Command)

(1) Alter the student table to make the following changes:

- ♠ Add a new column SEX which is of char datatype.
- ♠ Alter the column size of First_Name to 25.
- ♠ Alter the datatype of Dob to Date
- ♠ Add a primary key constraint for the Student Id.

(2) Alter the Course Table to make the following changes:

- ♠ Add a new column UNITS, which is of the number datatype.

(3) Alter the Instructor Table to make the following changes:

- ♠ Add a new column SEX which is of char datatype.

(4) Alter the new column Position which datatype varchar2 and size 25 with a check constraint which checks for

'ASSISTANT PROFESSOR','ASSOCIATE PROFESSOR','PROFESSOR'.

(5) Alter the Schedule Type Header Table to make the following changes:

1. Reduce the size of the schedule description to 100.
2. Add a primary key for Schedule_Id.

(6) Alter the Schedule Type Details to make the following changes:

- ♠ Add a composite primary key to the columns Schedule_id, day.
- ♠ Add a Foreign key to the column Schedule_id which refers schedule_id (Schedule Type Header).

(7) Alter the Class Location Table to make the following Changes:

- ♠ Alter the datatype of Location column to number.
- ♠ Add a composite primary key for (Class_Building, Class_Room)

(8) Alter the Class Table to make the following changes:

- ♠ Add a foreign key for the column Schdule_Id which refers schedule(Schedule_Id)
- ♠ Add a foreign key for the column Department_Id which refers Department(Department_Id)
- ♠ Add a foreign key for the column Course_Id which refers Course(Course_Id).

(9) Alter the Student_Sche to make the following changes:

- ♠ Add a composite primary key to the columns (Student_Id, Class_Id).
- ♠ Add a foreign key constraint to Student_Id which refers Student(Student_Id)
- ♠ Add a foreign key constraint to Class_Id which refers Class(Class_Id).

(10) Add the new column hod the data type is varchar2(25) into depart_info.

Lab 3

(1) Insert Records into the following tables. (Hint: insert minimum 10 records in each table).

1. Student Information Table.
2. Department Information Table.
3. Instructor Information Table.
4. Course Information Table.
5. Schedule Type Header Table.
6. Schedule Type Details Table.
7. Class Location Table.
8. Class Table.
9. Student Grade Information Table.

(2) Display all the inserted records in the each table.

(3) Set save points after every ten insertion.

(4) Roll back to the second save point.

(5) Set a save point upd.

(6) Update the records in the student information table.

(7) Set save point del.

(8) Delete records from the instructor table.

(9) Set save point ins.

(10) Insert a row in the course table.

(11) Roll back the transactions till the deletion.

Lab 4

(1) Enable the primary key student_id student table.

(2) Enable the primary key pk_course in course table.

(3) Disable the foreign key schedule_id in schedule_type_details table.

(4) Disable the foreign key fk_class_location in class table.

(5) Disable the foreign key fk_class_schedule_id in class table.

(6) Disable all foreign keys in student_sche table.

(7) Enable the foreign key fk_class_location in class table.

(8) Enable the foreign key schedule_id in schedule_type_details table.

(9) Enable the foreign key fk_class_schedule_id in class table.

(10) Enable all foreign keys in student_sche table.

(11) Drop the primary key pk_course course table.

(12) Drop the primary key student_id in student table.

(13) Drop the foreign key schedule_id in schedule_type_details.

(14) Add primary key to student table as before.

(15) Add primary key to course table as before.

(16) Add foreign key to schedule_id in the schedule_type_details as before.

Lab 5

(1) Display all information from the Student table whose lastname is null.

(2) Display the Student Id and the Firstname from the Student table who doesn't have a telephone and an email.

(3) Display Students Firstname whose city is Chennai.

(4) Display Students Lastname whose state starts with the letter "T".

(5) Display Students Id, LastName whose state ends with the letter 'A'.

(6) Display Students Firstname, Dob whose Firstname contains 'A' in the Fourth position.

(7) Display Students Firstname and Lastname Concatenated.

(8) Display all information from the Student table where the Students Firstname is of only ten characters.

(9) Display Students Firstname, Id and their age.

Lab 6

(1) Display Students Id whose age is greater than 20.

(2) Display Students Firstname, Dob whose Birthday falls today.

(3) Display the Students Firstname, Address whose birthday falls in the month of January.

(4) Display the eldest male Student's Firstname from the Student table.

(5) Display all information of the youngest female student from the Student table.

(6) Display the Students Id whose age is greater than 26 as on current date.

(7) Display the student's names whose first_name sounds alike.

(8) Display the Instructors Firstname, lastname concatenated together and the first character in capitals.

(9) Display the Student's firstname all in capitals.

Lab 7

(1) Display the last day of the current month.

(2) Display the first day of a given year.

(3) Display the last day of a given year.

(4) Display the no of months of the students since their birth.

(5) Display the next date of the given day that immediately follows the sysdate.

(6) Display the greatest date of the two given dates.

(7) Display the student's date of birth rounded to the nearest month.

(8) Display the Average age of the Students.

(9) Display the Students Firstname, Dob in the format "DD/MONTH/YYYY".

(10) Display number of unique cities from the Student table.

Lab 8

(1) Display Students Id from the Student table whose Lastnames are unique.

(2) Display Students Id, Firstname from the Student table whose Zipcode Contains two zeros.

(3) Display Course Id, Department Id from the Course table where the additional fees is the least.

(4) Display the Course Id, Additional fees from the course table with the additional fees in the format '$999.99'.

(5) Display the Average additional fees for the courses from the Course table.

(6) Display Additional fees from the course table rounded off to the nearest integer.

(7) Display minimum additional fees, maximum additional fees and difference between the minimum and the maximum fees from the course table.

(8) Display Scheduled Id from the Schedule Type table for which the starting time is next month and duration is less than three.

(9) Display Scheduled Id from the Schedule Type for which the starting time is current day.

(10) Display all from Schedule Type table where the starting time is between ten days prior the current date and ten days after the current date.

(11) Display Schedule Type, Duration, days past since their starting date from Schedule Type table.

(12) Display all from Schedule Type table with the month of starting time spelled out.

(13) Display all from Schedule Type table where the starting date falls in the current week of this month.

(14) Display all from the Class location table substitute the Seating capacity to 'Not Applicable" if its null.

(15) Display all from the Class table where the classrooms are unique and have the same instructor.

Lab 9

(1) Update all information's from the Student table whose lastname is null to a last name of 'Nil'.

(2) Update the Firstname from the Student table who doesn't have a telephone and an email to a value of 'Radiant'.

(3) Update Students Last name whose city is Chennai to 'Madrasi'.

(4) Update Students Last Name whose state starts with the letter 'T' to a value of 'TTT'.

(5) Update LastName whose state ends with the letter 'A' to a value of 'AAA'.

(6) Update Students Firstname to 'XXX', Dob to current date whose Firstname contains 'A' in the Fourth position.

(7) Update Students last name to 'YOUTH' whose age is less than 20.

(8) Update Students lastname to 'BIRTHDAY' whose Birthday falls today.

(9) Update the eldest male Student's Last name from the Student table to 'SENIOR'.

(10) Update students last name to 'JAN' whose birthday falls on January.

Lab 10

(1) Delete all information's from the Student table whose lastname is null.

(2) Delete the information from Student table that doesn't have a telephone and an email.

(3) Delete Students information whose city is Chennai.

(4) Delete Students information whose state starts with the letter 'T'

(5) Delete Students information whose state ends with the letter 'A'

(6) Delete Students information whose Firstname contains 'A' in the Fourth position.

(7) Delete all information from the Student table where the Students Firstname is of only ten characters.

(8) Delete Students information whose age is greater than 20.

(9) Delete Students information whose Birthday falls on June,05.

(10) Delete the Students information whose birthday falls in the month of October.

(11) Delete the eldest male Student's information from the Student table.

Lab 11

(1) Rename the student information table as student.

(2) Grant the alter, update, insert privileges to your friend on the table student.

(3) Revoke the above privileges.

(4) Create a savepoint s1.

(5) Insert two new student detail in student table.

(6) Rollback to savepoint s1.

(7) Commit the changes.

(8) Rename the student table as student information table.

(9) Create the employee table using instructors information table.

(10) Disable the primary key constraint of instructors information table.

(11) Truncate the employee table.

(12) Drop the foreign key constraint.

(13) Alter the table employee to add a primary key constraint on the new column age datatype raw. Is it possible to have a primary key on a column having datatype as raw? Can a table have more than one primary key?

(14) Drop the employee table.

Lab 12

(1) Display "MSU techno wing & CIT@MSU & RADIANT ray of hope" using concatenation operator.

(2) Concatenate and display the instructor's first and last name using built in function

(3) Create the employee table with following fields.

```
Empid          varchar2(4)
Joindate       date
Basic          number(10,2)
HRA            number(8,2)
DA             number(8,2)
PF             number(8,2)
```

(4) Insert the employee's details into the employee table.

(5) Display the employee's net salary using arithmetic operators.

(6) Calculate the simple interest using dual table.

(7) Using employee table display the Empid whose basic is equal to 3000.

(8) Display all the details of employee's whose basic is not equal to 4000.

(9) Display all the details of employee's whose HRA is greater than 750.

(10) Display the employee's details whose DA is less than 1000.

(11) Display the employee's record who's PF is lies between 500 to 1000 ranges.

(12) Display the employee's details whose id is either e101 or e103 or e104.

(13) Display the absolute value of the given number using built-in functions.

(14) Display the largest integer equal to or less than the given value using built-in functions.

(15) Display the smallest integer greater than or equal to given value using built-in functions.

(16) Find the total working days of employee's in employee table.

Lab 13

(1) Find the first name of the 'A' grade students.

(2) Find the duration of given schedule description.

(3) Find the additional fees of the given department name.

(4) Find the instructor first name of the given department id.

(5) Find the semester of given schedule description.

(6) Count the total number of students in student information table.

(7) Calculate the total duration per day in schedule table.

(8) Calculate the average basic of employee.

(9) Determine the minimum and maximum basic of employee; rename the title as max_basic and min_basic respectively.

(10) Count the number of employee having basic greater than or equal to 2000.

(11) Count the number of student whose telephone number is null.

(12) Count the number of student whose state is TN and august month babies.

(13) Concatenate the name of the student using built-in function.

(14) Translate the first name of the student information table 's first character 'a' as 'b'.

(15) Display the name of the instructor in upper case.

(16) Display the description of the schedule in lower case.

Lab 14

(1) Create a view named student from student information and department information tables that contains only the following columns student_id, first name, last name and department_id.

(2) Update the column of newly created view student. Observe the changes in the base tables.

(3) Create a synonym for course information table with name cours.

(4) Create a sequence instseq with the following specifications minimum value 1, maximum value 20, increment by 1, start with 0, with cycle and cache 10.

(5) Alter the sequence such that the maximum value is only 15.

(6) Create a local index named stud on first name of student information table.

(7) Create a unique index named stud1 on student_id column of the student information table.

(8) Create a view course_view from the course table with the following fields.

a. Course id.
b. Department id
c. Title
d. Description
e. Additional fees
f. Total fees (i.e. units *250 + additional fees)

(9) Create a view class_summary with the following fields and requirements.

a. Class id from class table
b. Class building from class table
c. Class room from class table
d. Department id from class table
e. Title from course table
f. Firstname from instructor table
g. Lastname from instructor table Provided,

Class.department id = course.department id
Class.course id = course.course id
Class.instructor id = instructor.instructor id

(10) Create a view dummy_view with columns (name, id) from the table(Dummy) which does not exist in the database.

(11) Create a table dummy with two columns(name, id).

(12) Create a view from the class table by substituting the column names with relevant names.

(13) Replace the view class_summary by removing the column title from the class table.

(14) Update the description of view course_view to null.

(15) Delete the rows from the course_view where deparment id is null and additional fees > 100 and total fees less than 500.

(16) Create a synonym class_alais for class table.

(17) Rename the synonym.

(18) Create a synonym for course_alais course table.

(19) Drop the synonym course_alais.

Lab 15

(1) Create an index stud_last on the student table (Lastname).

(2) Create an index instruct_first on the instructor table (firstname).

(3) Create an unique index dept_name on the department table(department name).

(4) Drop index instruct_first.

(5) Drop index stud_last.

(6) Drop index dept_name.

(7) Create a cluster named dept_cluster of datatype number(2).

(8) Create table dept using the dept_cluster with the following columns.

```
a. Deptno number(2)
b. Dname varchar2(30)
c. Loc varchar2(20)
```

(9) Create table emp using the dept_cluster with the following columns.

```
a. Empno number(3)
b. Ename varchar2(30)
c. Doj date
d. Desig varchar2(10)
e. Sex char(1)
f. Deptno number(2)
g. Salary number(7,2)
h. Comm number(7,2)
```

(10) Create a sequence named deptno_sequence; with starting value 1 and incrementing by 1, maximum value is 99, with cycle and cache.

(11) Create a sequence named empno_sequence. With starting value 100 and incrementing by 5, maximum value is 999, with no cycle and cache.

(12) Insert the deptno_sequence values into the dept table(deptno).

(13) Insert the empno_sequence values into the emp table(empno).

Lab 16

(1) Create a type address that would contain the following columns:

```
Column name     datatype
Add1                number(5)
Add2                varchar2(10)
```

```
Add3                varchar2(5)
Add4                number(7)
```

(2) Create another type emp which will have the above defined user type and the following columns:

```
Column name     data type
Eno             number(2)
Ename           varchar2(10)
Eadd                address
```

(3) Insert into the object emp values for the columns eno as 100 and eadd as 100, first st, Chennai, 600078.

(4) Insert into the object emp values for columns eno as 101 and eadd as 102, iii phases, bgl, 576897.

(5) Write a query to display the eno and add1 from the emp object where add1 has a value of 11.

(6) Write a query to update the eadd column to be 10, ii st, mds,35567 where the eno is 100.

(7) Write a query to delete from the emp table that row whose add1 has a value of 10.

(8) Create a type named personal whose structure is given below:

```
Column name     data type
Name                varchar2(20)
Age             number(3)
Sex             char(1)
```

(9) Create a table whose structure is as shown below:

```
Column name     data type
Id              number(4)
Centre          varchar2(15)
Per             personal
```

(10) Insert into the above table the values given:

```
1234,Chennai, (meena, 22,f)
2345,nellai, (veena, 23,f)
3456,Bangalore, (reena, 24,f)
```

(11) Drop the type personal that has been created.

(12) Create a type person_details with structure as shown below:

```
Column name     data type
Name            varchar2(15)
Age             number(2)
Desg            varchar2(15)
```

(13) Create a table person_details_table_ty which is based on the type created above.

(14) Create another table other_info_person, has a structure as shown below. It should also contain a stored table named as person_store_table.

```
Column name     data type
Dept_name       varchar2(10)
Dept_no         number(3)
Person_info     person_details_table_ty
```

(15) Create a varying array which will contain a maximum of 3 rows and will be of data type varchar2(8)

(16) Create a type dept_type whose structure is given below:

```
Column name     data type
Deptno          number(2)
Dname           varchar2(14)
Loc             varchar2(13)
```

(17) Create a table dept_t which consists of the above type.

(18) Create a table emp which will hold a reference to the above created table and also have a structure as given below:

```
Column name     data type
Empno           number(4)
Job             varchar2(10)
Mgr             number(4)
Hiredate        date
Sal             number(7,2)
Comm            number(7,2)
Dept            ref dept_type
```

Lab 17

(1) Write PL/SQL block to increase the salary by 10% if the salary is > 2500 and > 3000.

(2) Write PL/SQL block to decrease the salary by 13% if the salary is >3000 and < 5000.

(3) Write PL/SQL block to increase the salary by 15% if the salary is > 5000.

While Loops.
(Hint: use basic loops, while loops, numeric for loops, cursor for loops.)

(4) Write PL/SQL block to insert student details into student table until the user wishes to stop.

(5) Write PL/SQL block to insert department details into Department table until the user wishes to stop.

(6) Write PL/SQL block to display the names of those employees getting salary > 3000.

(7) Write PL/SQL block to display the male employees details.

(8) Write PL/SQL block to display the total salary (I,e.Salary +Comm) of each employee whose comm Is not null.

(9) Write PL/SQL block to insert the instructor details from the instructor table with the record count until record count is > 20.

(10) Write PL/SQL block to display the total salary (I,e .Salary + Comm.) of each employee whose comm. Is not null until sum of total salary exceeds 25,000.

(11) Write PL/SQL block to display the employee details from the employee table, and get the user's choice and delete the record if the user wishes to delete.

(12) Write PL/SQL block to insert only the odd numbers from 1 to 20 into department table as department id and get the department name from the user.

Lab 18

Cursors.

Note: use the cursor attributes % found, % notfound, % rowcount, % isopen

(1) Write PL/SQL block to increase the salary by 15 % for all employees in emp table.

(2) Write PL/SQL block to decrease the additional _fees in the Course table to 5%.

(3) Write PL/SQL block to increase the additional fees by 10% and if the additional fees exceeds 100 then decrease the additional fees by 20%.

(4) Write PL/SQL block to display the employee details, and the number of employees joined in each month.

(5) Write a PL/SQL block to get a class_room from the user and display the Starting_time, Duration from the schedule_type table for that room.

(6) Write a PL/SQL block to get a instuctor first_name and class_room from the user and display the instructor details, schedule_type details. From the class, schedule_type instructor tables.

(7) Write a PL/SQL block to display the schedule_id, schedule_description, day, starting_time, and duration from the schedule_type header, schedule_type details tables.

(8) Create a Pl/SQL tables containing the instructor_id, first_name, last_name of the instructor table and insert values into the table.

(9) Delete all he rows from the PL/SQL tables.

(10) Create a PL/SQL tables containing class_id, course_id, department_id, instructor_id, instructor first_name of the class, instructor tables and insert values into the tables.

Lab 19

Exceptions

(1) Write PL/SQL block to handle the exception no_data_found for the question 5,6 given above.

(2) Write PL/SQL block to handle the exception dup_val_on_index by inserting a duplicate row in the course table.

(3) Write PL/SQL block to handle the exception value_error by inserting a value of width greater than 20 into the department table.

(3) Write PL/SQL block with a user defined exception and raise the exception, and handle the exception.

(4) Write PL/SQL block to handle exception, which are not handled by, using others and display a message.

Sub programs & packages

(1) Create a procedure that takes an argument (description) and deletes the row from the course table.

(2) Create a procedure that displays the instructor details, class details and the student details of a particular student which the user inputs.

(3) Write a function that takes two arguments viz. student_id, class_id from the user and checks whether any conflict occurs with any others class with the current class schedule. If occurs give an alert message that a conflict occurs with the corresponding class else display no conflict.

(4) Write a function that will get the value from the user and insert rows into the class table without any violation of the integrity constraints.

(5) Write a function that will display the constraint details in the given table.

(6) Create a package that contains overloaded functions for

a. Adding five integers
b. Subtracting two integers
c. Multiplying three integers

(7) Create a package that contains the following;

i. Function to register for class.
Arguments (student_id, class_id)

Check if the student has already registered or not.
Check for conflict in class schedules.

ii. Function to check for conflict in classes.
Arguments (student_id, class_id)

iii. Procedure to assign an instructor to a class.

Arguments (class_id, instructor_id)

Check if the instructor is associated with the department that offers the class.

iv. Procedure to assign grade.
Arguments (student_id, class_id , grade)

Check if the students is registered for the class.

v. Function to calculate the average grade point for the given student.
Arguments (student_id)

Lab 20

Database Triggers.

(1) Write a database trigger before insert for each row on the course table not allowing transactions on Sundays and Saturdays.

(2) Write a database trigger after update for each row giving the date and the day on which the update is performed on the class table.

(3) Write a database trigger before delete for each row not allowing deletion and giving message on the department table.

(4) Write a database trigger after delete for each row on instructor table, which creates a new table named dump for the first delete, and inserts the deleted row into that table.

(5) Write a database trigger before insert/delete/update for each row not allowing any of these operations on the table student on Mondays. Wednesdays, Sundays.

(6) Write a database trigger before insert/delete/update for each row not allowing any of these operations on the table instructor between 6.p.m to 10 a.m

Lab 21

(1) Create a PL/SQL function compute with two parameters eno and bonus. Using this function create a java program and call this function.

(2) Create a java program using sql query. Select the salary of given employee number.

(3) Create a java program using #sql command retrieves the employee's details. Import the sqlj tool.

Lab 22

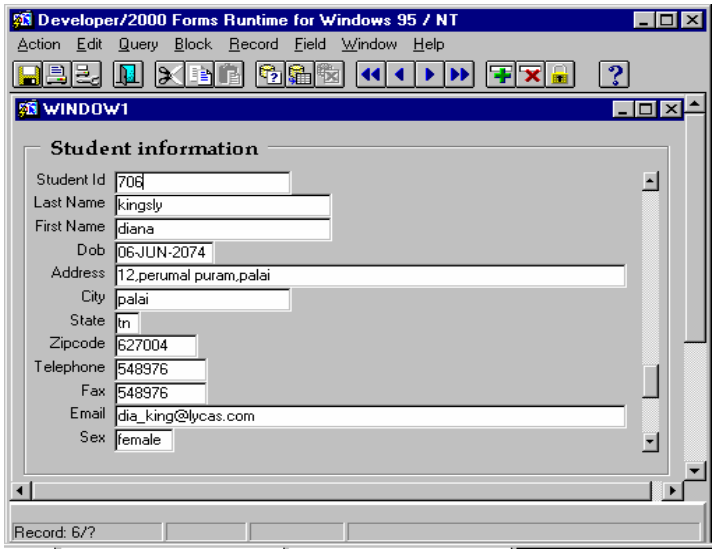
(1) Create a java program insert the records into the student table. Import the sqlj tool.

(2) Create a java program insert the records into the depart table using PL/SQL commands. Import the sqlj tool.

(3) Create a java program select the record from emp table and display the records.

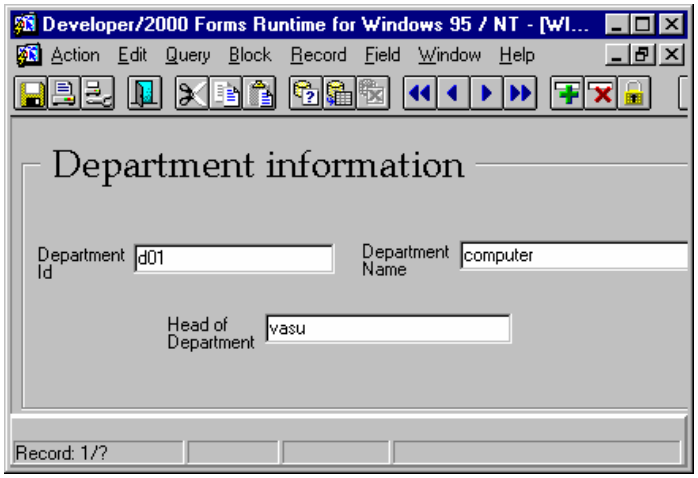
Lab 23

(1) Create a new form module for student information table. Create a new block by using the data block wizard.

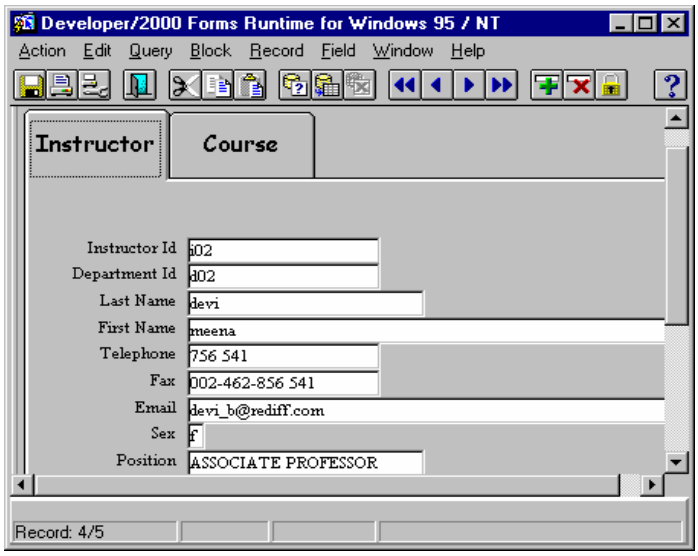


Run your form module. Execute a query. Exit run time and return to form builder.

(2) Create new form department information using data block wizard. And modify layout wizard and run the form module. Execute a query. Exit run time and return to form builder.

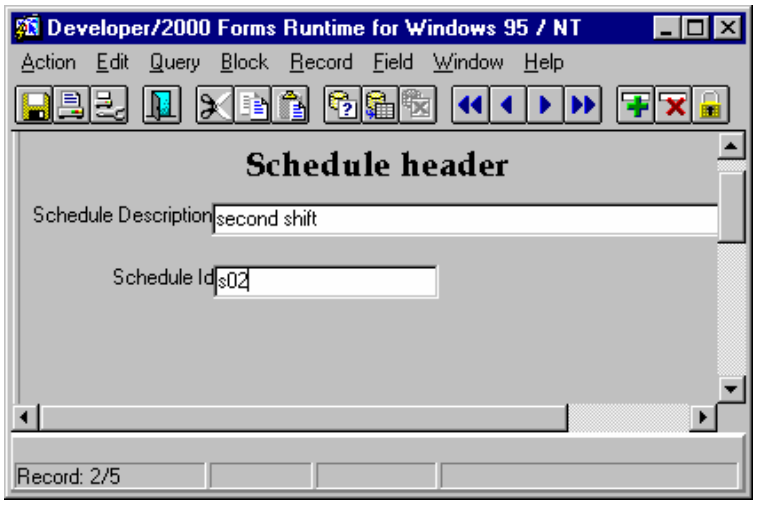


(3) Create new form for instructor and course table using tab canvas.

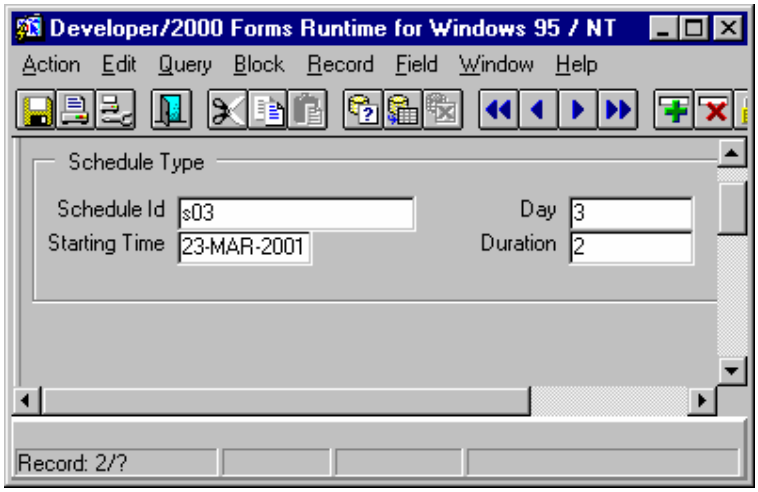


Lab 24

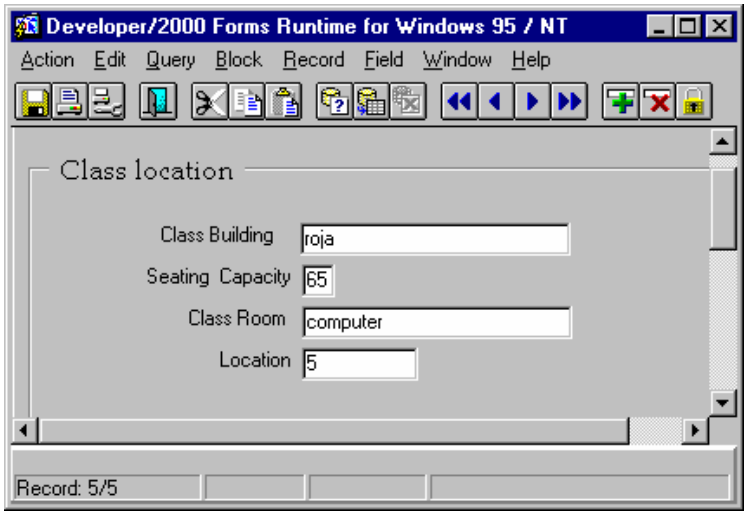
(1) Create new form for schedule using stacked canvas, run the form, execute query, and exit the form.



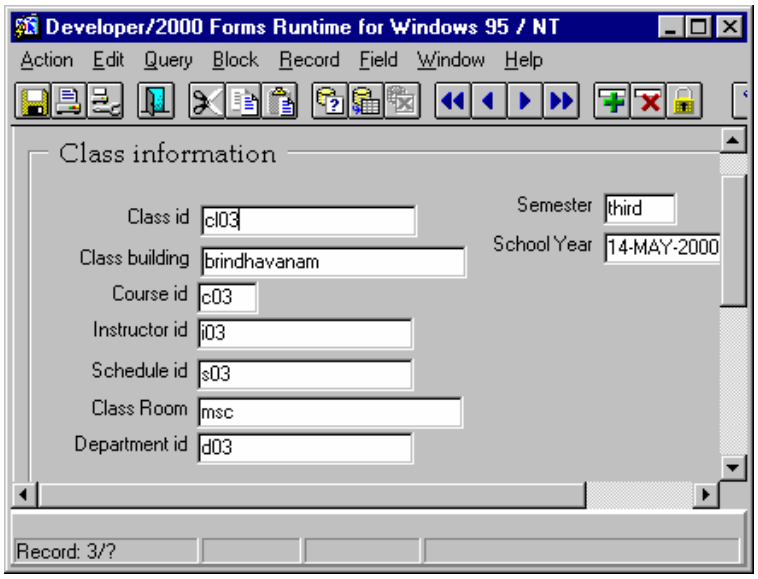
(2) Create a form for schedule type. Using data block wizard.



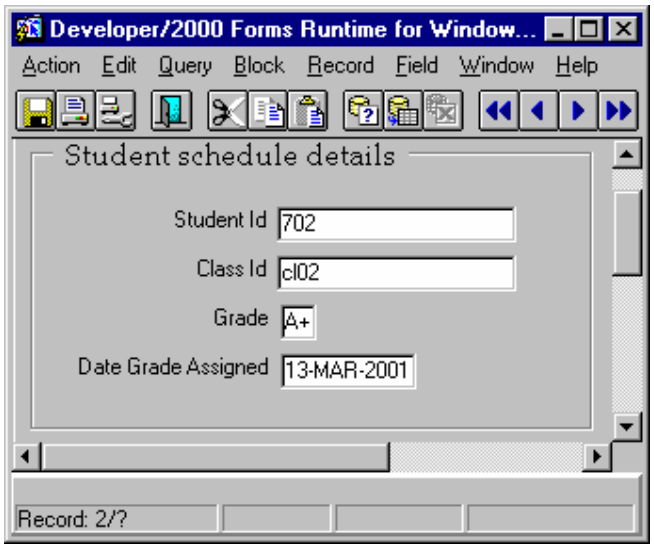
(3) Create a form for class location using content canvas.



(4) Create a form for class table. Run the form and execute the query insert one new record and save it.

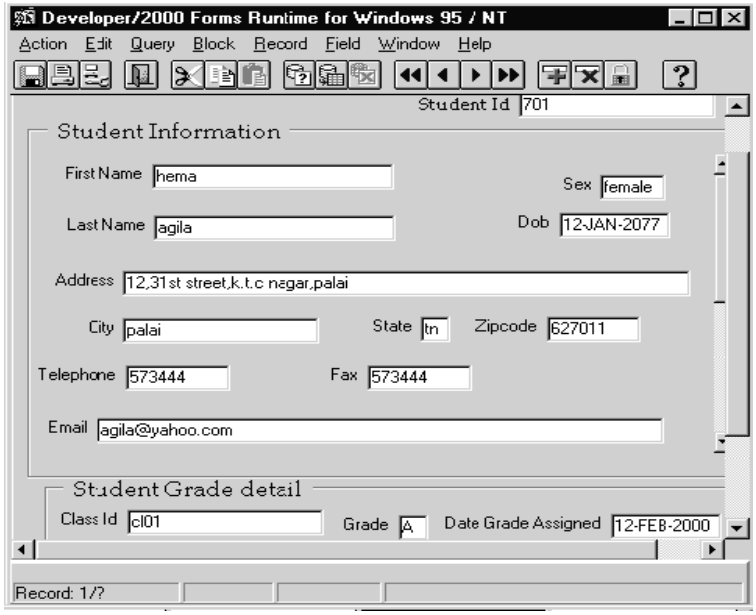


(5) Create a form for student grade details. Run a form and delete one record and change the student grade details.

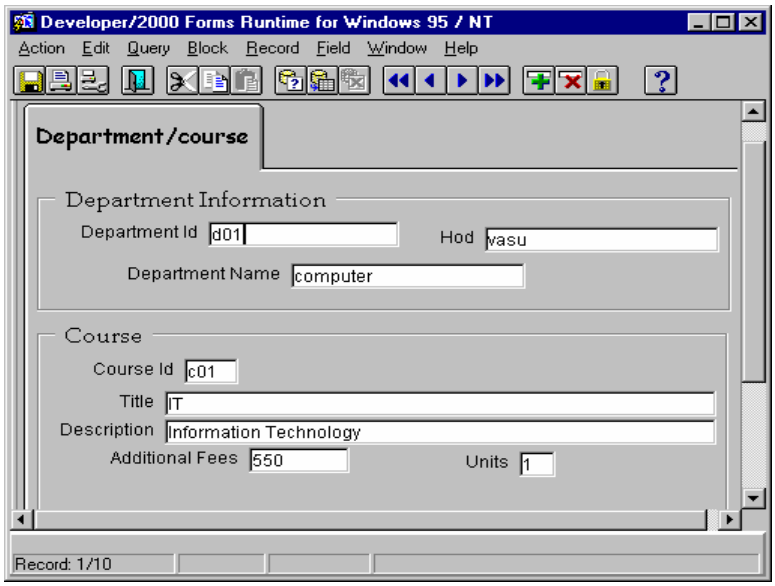


Lab 25

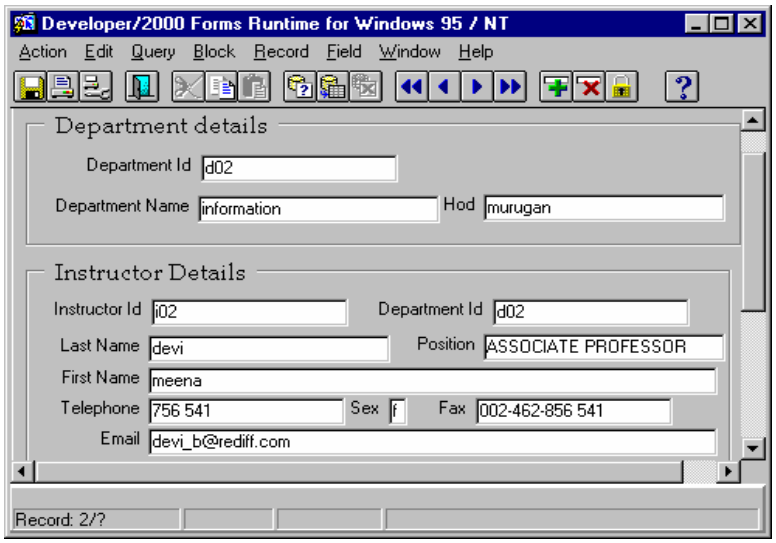
(1) Create master detail relations for student and student grade table. Student information table is the master table and student grade table is the detail table it gives the grade detail of the course.



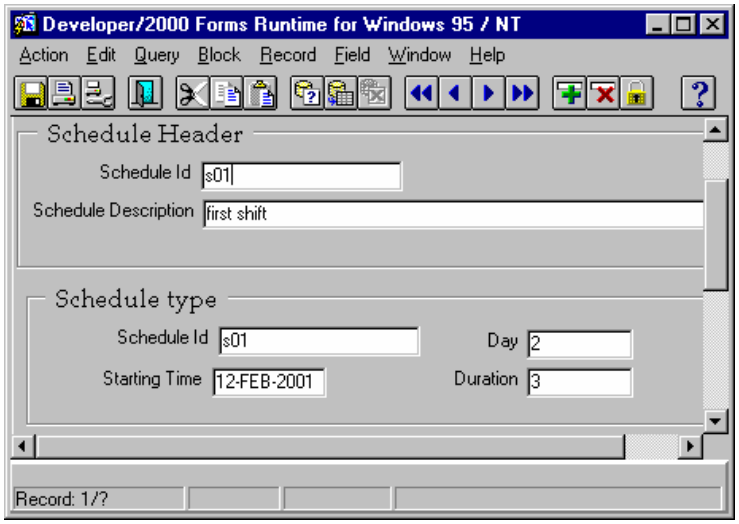
(2) Create a new block by using the data block wizard. Base the block on the course table and include all columns. Create a relationship and select the master block as department id. Display all items except department id on the course table. Display the record in this detail block on the same canvas as the master block. Use a tabular style layout and include a scroll bar.



(3) Create a master detail form for department and instructor tables.

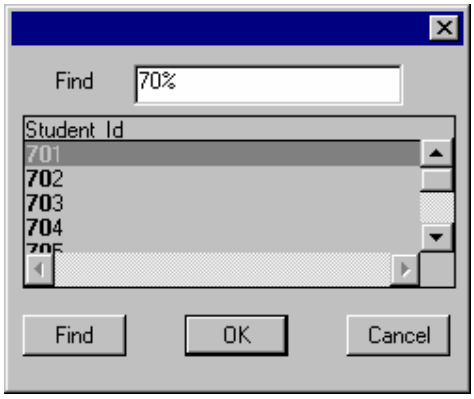


(4) Create a master detail relation for schedule header and schedule type tables.

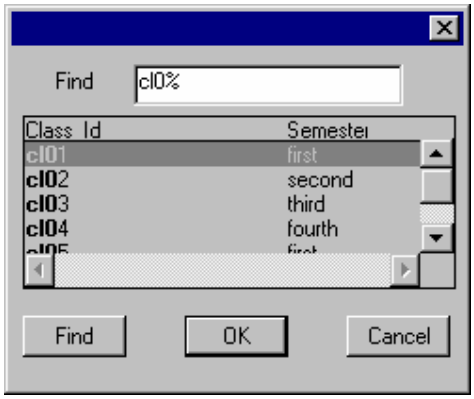


Lab 26

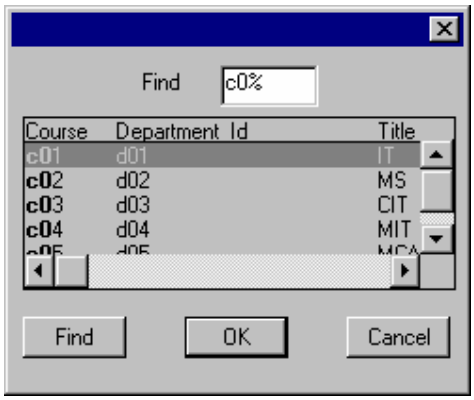
(1) In this lab session, you will create two LOVs and an editor. Using the LOV wizard, create an LOV in the student information form to display student id. Attach the LOV to the student_id item in the data block. Save and run the form.



(2) Using the LOV wizard, create an LOV in the class form. Attach the LOV to the class_id in the data block. Save and run the form.



(3) Using the LOV wizard, create an LOV in the course form. Attach the LOV to the course_id in the data block. Save and run the form.



Lab 27

(1) Create a form to calculate the bonus details for the employee's with the following fields ecode, ename, designation, salary and bonus. Using radio group control checks the designation of the

employee (like manager 40%, analyst 30%, programmer 20%, clerk 10%, technical writer 5%). When we click one button it will calculate the bonus of the employee for the selection of designation.

(2) Create a form for telephone bill calculation with the following fields cname, phno, address, loccall, stdcall, stdcharge and totalch. Calculate the stdcharge and total charge.

(3) Create a railway ticket reservation form for using reservation table trno, date_of_journey, class, no.of_seats, berth/seat, start_place, destination_place and train_fare. Calculate the total amount for reserved seats per classes and display it in display item. Insert the new instance using add button. Include the modify for update the records, cancel for delete the records, calculate for total train_fare calculation (no.of_seats * train_fare). And include the navigation button also for moving records.

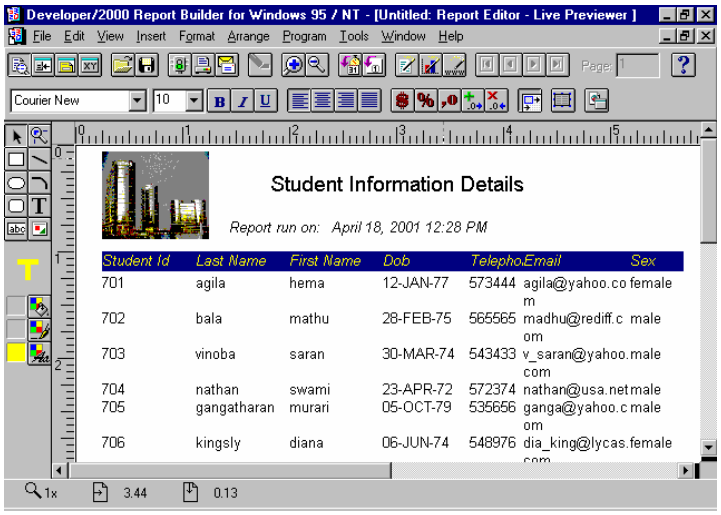
Lab 28

(1) Create a form for maintaining the stock details. Using the following records pid, pname, unit_price, qty, costprice and sellingprice. Calculate the total price for purchasing products and display it in one display item.

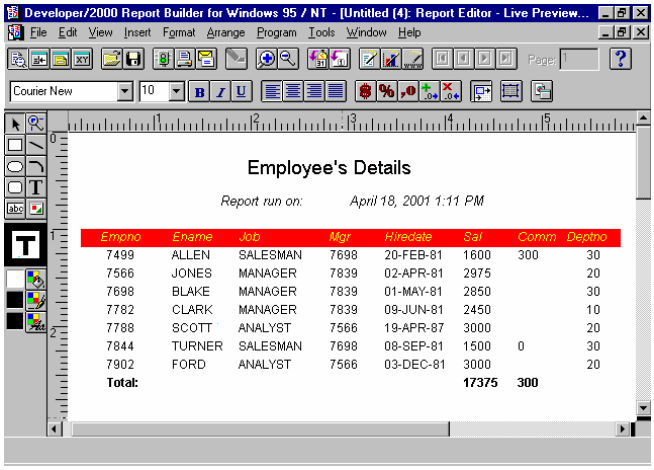
(2) Create a form for calculate the charge of patient details. Calculate both in-patient and out-patient details. Using patient table in this table the columns are pat_id, name, age, pat_cat, diseases, consult_fees, inject_fees, operation_fees, rent, medicine and ECG scan. In this details calculate patient total charge using patient category manner the patient category is 'in' and 'out'. If the patient is 'out' category then calculate the total charge using consult_fees, inject_fees and medicine but the patient is 'in' then calculate the total charge using consult_fees, inject_fees, rent, medicine if there is ECG scan or operation_fees then include these fees also and calculate the total charge for in-patient. Create menu for this form in this menu include add, save, edit, delete, view, navigation menu (first, next, previous, last) and exit.

Lab 29

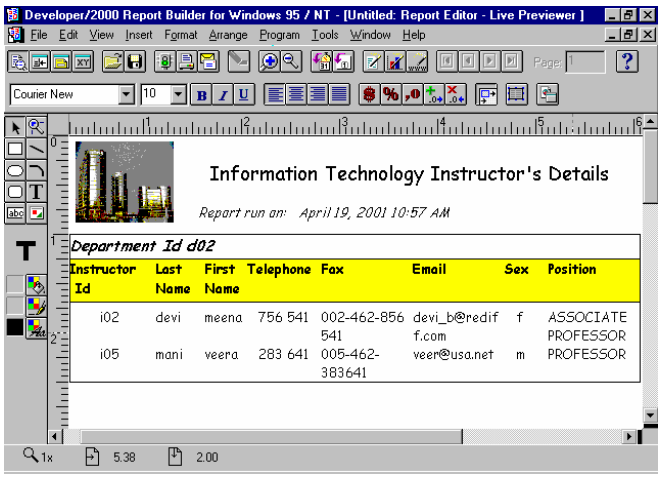
(1) Create a tabular report for student information table.



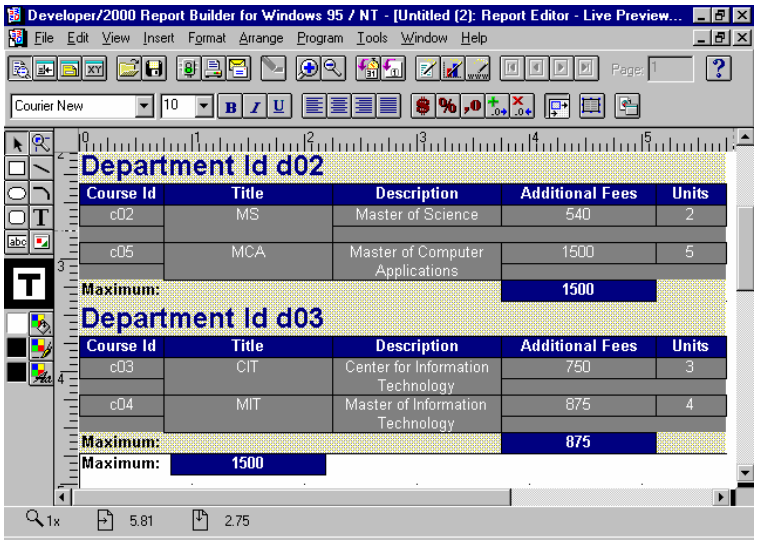
(2) Create a tabular form report for employee table and calculate the total employee's salary and their commission.



(3) Create a report for display the information technology department instructor's only.

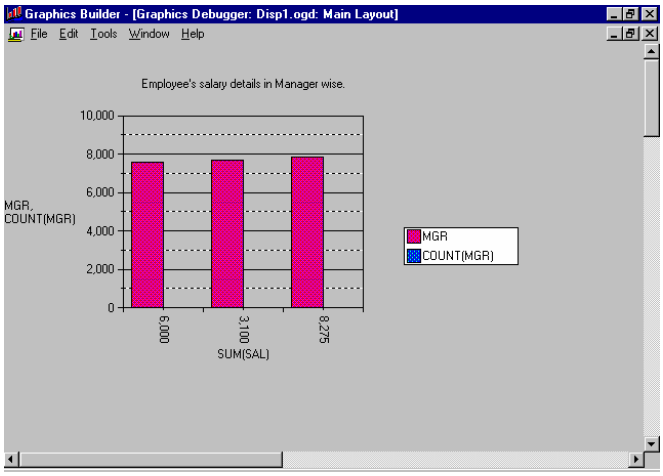


(4) Create a report for group the course table in department wise and find the maximum fees for each department.

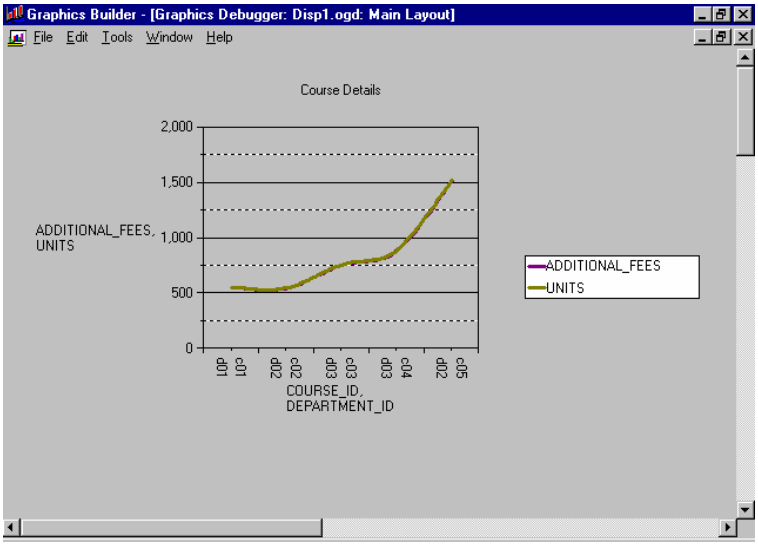


Lab 30

(1) Create a graph for employee's salary details grouping it with manager wise.



(2) Create a graph for course details.



2.1 RDBMS concepts and ORACLE8i

Lab 1

1. create table stud_info(Student_id varchar2(20), Last_name varchar2(25),First_name varchar2(20), Dob varchar2(20), Address varchar2(300), City varchar2(20),State varchar2(2), ZipCode varchar2(9),Telephone varchar2(10), Fax varchar2(10), Email varchar2(100));

   Table created.

2. create table depart_info(Department_Id varchar2(20)primary key, Department_Name varchar2(25));

   Table created.

3. create table inst(Instructor_id varchar2(20) primary key, Department_Id varchar2(20) references depart_info(department_id),Last_Name varchar2(25), First_Name varchar2(200),Telephone varchar2(20), Fax varchar2(20),Email varchar2(100));

   Table created.

4. create table course(Course_Id varchar2(5),Department_Id varchar2(20) references depart_info(department_id),Title char(60),Description varchar2(200), Additional_fees number primary key);

   Table created.

5. create table sche_head(Schedule_Id varchar2(20) primary key,Schedule_Description varchar2(200));

   Table created.

6. create table sche_type(Schedule_Id varchar2(20),Day number,Starting_Time date,Duration number);

   Table created.

7. create table class_loc(Class_Building varchar2(25), Class_Room varchar2(25), Seating_Capacity varchar2(2));

   Table created.

8.
```
create table class(Class_Id varchar2(20) primary key,
Schedule_Id varchar2(20),Class_Building varchar2(25),
Class_Room varchar2(25),Course_Id varchar2(5),
Department_Id varchar2(20),Instructor_Id varchar2(20)
references Inst(Instructor_Id),Semester varchar2(6), School_Year date);
```

```
Table created.
```

9.
```
create table stud_sche(Student_Id varchar2(20),
Class_Id varchar2(20),Grade varchar2(2) check (Grade
in('A','A+','A-','B','B+','B-','C','C+','C-','D','D+','D-','F','F+','F-')),
Date_Grade_Assigned date);
```

```
Table created.
```

10.
```
Desc stud_info
```

```
Name                           Null?    Type
------------------------------ -------- ----
STUDENT_ID                              VARCHAR2(20)
LAST_NAME                               VARCHAR2(25)
FIRST_NAME                              VARCHAR2(20)
DOB                                     VARCHAR2(20)
ADDRESS                                 VARCHAR2(300)
CITY                                    VARCHAR2(20)
STATE                                   VARCHAR2(2)
ZIPCODE                                 VARCHAR2(9)
TELEPHONE                               VARCHAR2(10)
FAX                                     VARCHAR2(10)
EMAIL                                   VARCHAR2(100)
```

11.
```
desc depart_info
```

```
Name                           Null?    Type
------------------------------ -------- ----
DEPARTMENT_ID                  NOT NULL VARCHAR2(20)
DEPARTMENT_NAME                         VARCHAR2(25)
```

12. desc inst

```
Name                           Null?    Type
------------------------------ -------- ----
INSTRUCTOR_ID                  NOT NULL VARCHAR2(20)
DEPARTMENT_ID                           VARCHAR2(20)
LAST_NAME                               VARCHAR2(25)
FIRST_NAME                              VARCHAR2(200)
TELEPHONE                               VARCHAR2(20)
FAX                                     VARCHAR2(20)
EMAIL                                   VARCHAR2(100)
```

13. desc course

```
Name                           Null?    Type
------------------------------ -------- ----
COURSE_ID                               VARCHAR2(5)
DEPARTMENT_ID                           VARCHAR2(20)
TITLE                                   CHAR(60)
DESCRIPTION                             VARCHAR2(200)
ADDITIONAL_FEES                NOT NULL NUMBER
```

14. desc sche_head

```
Name                           Null?    Type
------------------------------ -------- ----
SCHEDULE_ID                    NOT NULL VARCHAR2(20)
SCHEDULE_DESCRIPTION                    VARCHAR2(200)
```

15. desc sche_type

```
Name                           Null?    Type
------------------------------ -------- ----
SCHEDULE_ID                             VARCHAR2(20)
DAY                                     NUMBER
STARTING_TIME                           DATE
DURATION                                NUMBER
```

16. desc class_loc

```
Name                           Null?    Type
------------------------------ -------- ----
CLASS_BUILDING                          VARCHAR2(25)
CLASS_ROOM                              VARCHAR2(25)
SEATING_CAPACITY                        VARCHAR2(2)
```

17. desc class

```
Name                           Null?    Type
------------------------------ -------- ----
CLASS_ID                       NOT NULL VARCHAR2(20)
SCHEDULE_ID                             VARCHAR2(20)
CLASS_BUILDING                          VARCHAR2(25)
```

```
CLASS_ROOM                                VARCHAR2(25)
COURSE_ID                                 VARCHAR2(5)
DEPARTMENT_ID                             VARCHAR2(20)
INSTRUCTOR_ID                             VARCHAR2(20)
SEMESTER                                  VARCHAR2(6)
SCHOOL_YEAR                               DATE
```

18. desc stud_sche

```
Name                           Null?    Type
------------------------------ -------- ----
STUDENT_ID                               VARCHAR2(20)
CLASS_ID                                 VARCHAR2(20)
GRADE                                    VARCHAR2(2)
DATE_GRADE_ASSIGNED                      DATE
```

Lab 2

1. ♠ alter table stud_info add(sex char(6));

   Table altered.

   ♠ alter table stud_info modify(first_name varchar2(25));

   Table altered.

   ♠ alter table stud_info modify(dob date);

   Table altered.

   ♠ alter table stud_info add primary key(student_id);

   Table altered.

2. alter table course add(units number(2));

   Table altered.

3. alter table inst add(sex char);

   Table altered.

4. alter table inst add(position varchar2(25)check(position in('ASSISTANT PROFESSOR','ASSOCIATE PROFESSOR','PROFESSOR')));

   Table altered.

5. ♠ alter table sche_head modify(schedule_description varchar2(100));

   Table altered.

   ♠ alter table sche_head add primary key(schedule_id);

   Table altered.

6. ♠ alter table sche_type add primary key(day);

   Table altered.

   ♠ alter table sche_type add foreign key(schedule_id)
   references sche_head(schedule_id);

   Table altered.

7. ♠ alter table class_loc add(location number);

   Table altered.

   ♠ alter table class_loc add primary key (class_building,
class_room);

   Table altered.

8. ♠ alter table class add foreign key(schedule_id) references
   sche_head(schedule_id);

   Table altered.

   ♠ alter table class add foreign key(department_id)
   references depart_info(department_id);

   Table altered.

   ♠ alter table class add foreign key(course_id) references
   course(course_id);

   Table altered.

9. ♠ alter table stud_sche add primary key (student_id, class_id);

   Table altered.

   ♠ alter table stud_sche add foreign key(student_id) references
   stud_info(student_id);

   Table altered.

   ♠ alter table stud_sche add foreign key(class_id) references
   class(class_id);

   Table altered.

10. alter table depart_info add(hod varchar2(25));

    Table altered.

Lab 3

1. Insert into student_info values('&Student_id','&Last_name',

```
'&First_name','&Dob', '&Address', '&city', '&State', '&ZipCode',
'&Telephone', '&Fax', '&Email','&sex');

Enter value for student_id: 701
Enter value for last_name: agila
Enter value for first_name: hema
old   1: insert into student_info values('&Student_id', '&Last_name',
'&First_name',
new   1: insert into student_info values('701','agila','hema',
Enter value for dob: 12-jan-77
Enter value for address: 12,31st street, k.t.c nagar, palai
Enter value for city: palai
Enter value for state: tn
Enter value for zipcode: 627011
old   2: '&Dob', '&Address', '&city', '&State', '&ZipCode',
new   2: '12-jan-77', '12,31st street, k.t.c nagar, palai', 'palai', 'tn',
'627011',
Enter value for telephone: 573444
Enter value for fax: 573444
Enter value for email: agila@yahoo.com
Enter value for sex: female
old   3: '&Telephone', '&Fax', '&Email','&sex')
new   3: '573444', '573444', 'agila@yahoo.com','female')

1 row created.
```

2.
```
insert into depart_info values('&department_id',
'&department_name','&hod');

Enter value for department_id: d01
old   1: insert into depart_info values('&department_id',
new   1: insert into depart_info values('d01',
Enter value for department_name: computer
Enter value for hod: vasu
old   2: '&department_name','&hod')
new   2: 'computer','vasu')

1 row created.
```

3.
```
insert into inst values ('&INSTRUCTOR_ID', '&DEPARTMENT_ID',
'&LAST_NAME', '&FIRST_NAME','&TELEPHONE','&FAX','&EMAIL',
'&SEX', '&POSITION');

Enter value for instructor_id: i01
Enter value for department_id: d01
Enter value for last_name: bala
old   1:  insert into inst
values('&INSTRUCTOR_ID','&DEPARTMENT_ID','&LAST_NAME',
new   1:  insert into inst values('i01','d01','bala',
Enter value for first_name: madhu
Enter value for telephone: 271 584
Enter value for fax: 001-462-371 584
Enter value for email: bala@yahoo.com
Enter value for sex: m
```

```
   old   2:  '&FIRST_NAME','&TELEPHONE','&FAX','&EMAIL','&SEX',
   new   2:  'madhu','271 584','001-462-371 584','bala@yahoo.com','m',
   Enter value for position: PROFESSOR
   old   3:  '&POSITION')
   new   3:  'PROFESSOR')

   1 row created.

4. into course values('&course_id','&DEPARTMENT_ID',
   '&TITLE','&DESCRIPTION',&ADDITIONAL_FEES,&UNITS)

   Enter value for course_id: c01
   Enter value for department_id: d01
   old   1: insert into course values('&course_id','&DEPARTMENT_ID',
   new   1: insert into course values('c01','d01',
   Enter value for title: IT
   Enter value for description: Information Technology
   Enter value for additional_fees: 550
   Enter value for units: 1
   old   2: '&TITLE','&DESCRIPTION',&ADDITIONAL_FEES,&UNITS)
   new   2: 'IT','Information Technology',550,1)

   1 row created.

5. into sche_head values('&SCHEDULE_ID',
   '&SCHEDULE_DESCRIPTION');

   Enter value for schedule_id: s01
   old   1: insert into sche_head values('&SCHEDULE_ID',
   new   1: insert into sche_head values('s01',
   Enter value for schedule_description: first shift
   old   2: '&SCHEDULE_DESCRIPTION')
   new   2: 'first shift')

   1 row created.

6. SQL> insert into sche_type
   values('&schedule_id',&day,'&start_time', 2  &duration);

   Enter value for schedule_id: s01
   Enter value for day: 2
   Enter value for start_time: 12-aug-77
   old   1: insert into sche_type values('&schedule_id',&day,'&start_time',
   new   1: insert into sche_type values('s01',2,'12-aug-77',
   Enter value for duration: 3
   old   2: &duration)
   new   2: 3)

   1 row created.

7. SQL> insert into class_loc
   values('&class_building','&class_room',
   2 '&seating_capacity',&location);
```

```
Enter value for class_building: nandhavanam
Enter value for class_room: commerce
old   1: insert into class_loc values('&class_building','&class_room',
new   1: insert into class_loc values('nandhavanam','commerce',
Enter value for seating_capacity: 53
Enter value for location: 01
old   2: '&seating_capacity',&location)
new   2: '53',01)

1 row created.
```

8.
```
insert into class values('&CLASS_ID','&SCHEDULE_ID',

'&CLASS_BUILDING','&CLASS_ROOM','&COURSE_ID','&DEPARTMENT_ID',
'&INSTRUCTOR_ID','&SEMESTER','&SCHOOL_YEAR');

Enter value for class_id: cl02
Enter value for schedule_id: s02
old   1:  insert into class values('&CLASS_ID','&SCHEDULE_ID',
new   1:  insert into class values('cl02','s02',
Enter value for class_building: nandhavanam
Enter value for class_room: mca
Enter value for course_id: c02
Enter value for department_id: d01
old   2:

'&CLASS_BUILDING','&CLASS_ROOM','&COURSE_ID','&DEPARTMENT_ID',
new   2:  'nandhavanam','mca','c02','d01',
Enter value for instructor_id: i01
Enter value for semester: second
Enter value for school_year: 12-apr-00
old   3:  '&INSTRUCTOR_ID','&SEMESTER','&SCHOOL_YEAR')
new   3:  'i01','second','12-apr-00')

1 row created.
```

9.
```
insert into stud_sche values('&STUDENT_ID','&CLASS_ID',
'&GRADE','&DATE_GRADE_ASSIGNED')

Enter value for student_id: 701
Enter value for class_id: cl01
old   1: insert into stud_sche values('&STUDENT_ID','&CLASS_ID',
new   1: insert into stud_sche values('701','cl01',
Enter value for grade: A
Enter value for date_grade_assigned: 12-feb-00
old   2: '&GRADE','&DATE_GRADE_ASSIGNED')
new   2: 'A','12-feb-00')

1 row created.

2.1 select * from student_info;
```

```
2.2 Select * from depart_info;

DEPARTMENT_ID        DEPARTMENT_NAME           HOD
-------------------- ------------------------- -------------
d01                  computer                  vasu
d02                  information               murugan
d03                  biochemistry              esakki
d04                  microbiology              manikkam
d05                  english                   kavitha
d06                  arts&science              kumar
d07                  commerce                  arasu
d08                  economic                  saravanan
d09                  tamil                     sankar
d10                  sports                    marthandam

10 rows selected.

2.3 select * from inst;

2.4 select * from sche_head;

2.5 select * from sche_type;

2.6 select * from class_loc;

2.7 select * from class;

2.8 select * from course;

2.9 select * from stud_sche;
```

3. savepoint s1;

4. rollback;

5. savepoint upd;

6. update stud_info set telephone = ‘333444’ where student_id = ‘705’;

7. savepoint del;

8. delete from inst where instructor_id =’i01’;

9. savepoint ins;

10. insert into course values('c07','d04', 'MFM ',' Master of finance management',500,5);

11. rollback;

Lab 4

1. Alter table stud_info enable stud_pk;

```
2. Alter table course enable pk_course;

3. Alter table sche_type disable sche_pk;

4. Alter table class disable fk_class_location;

5. Alter table class disable fk_class_schedule_id;

6. Alter table stud_sche disable stud_fk;

7. Alter table stud_sche disable class_fk;

8. Alter table class enable fk_class_location;

9. Alter table sche_type enable sche_fk;

10. Alter table class enable fk_class_schedule_id;

11. Alter table stud_sche enable stud_fk;

12. Alter table stud_sche enable class_fk;

13. Alter table course drop pk_course;

14. Alter table stud_info drop stud_pk;

15. Alter table sche_type drop sche_fk;

16. Alter table stud_info add primary key(student_id);

17. Alter table course add primary key(course_id);

18. Alter table sche_type add foreign key(schedule_id) references
    sche_head(schedule_id);
```

Lab 5

```
1. Select * from stud_info where Lastname is null;

2. Select student_id, Firstname from stud_info where telephone is  null and
email is null;

3. Select firstname from stud_info where city = 'chennai';


4. Select Lastname from stud_info where state like 'T%';

5. Select student_id, Lastname from stud_info where state like  '%a';

6. Select Firstname, dob from stud_info where Firstname like '___a%';

7. Select concat(Firstname, Lastname) from stud_info;
```

```
8. Select * from stud_info where length(first_name) = 10;

9. select first_NAME,student_id,to_char(sysdate,'yyyy')-   to_char(dob,'yyyy')
as age from stud_info;
```

Lab 6

```
1. Select student_id from stud_info where to_char(sysdate,    'yyyy')-
to_char(dob,'yyyy') > 20;

2. Select First_name, dob from stud_info where to_char(dob,  'month') =
to_char(sysdate,'month') and to_char(dob,'dd')= to_char(sysdate,'dd');

3. Select First_name, address from stud_info where to_char(dob, 'month')=
'january' and to_char(dob,'day')='monday';

4. Select first_name from stud_info where dob =(select
   Min(dob) from stud_info) and sex='male';

5. Select * from stud_info where dob =(select Max(dob) from   stud_info) and
sex='female';

6. Select student_id from stud_info where to_char(sysdate,    'yyyy')-
to_char(dob,'yyyy') > 26;
7. Select first_name from stud_info where
   soundex(first_name)=Soundex(first_name);

8. Select initcap(concat(first_name,last_name)) from stud_info;

9. Select Upper (first_name) from stud_info;
```

Lab 7

```
1. Select last_day(sysdate) from dual;

2. Select round(sysdate,'day') from dual;

3. Select trunc(sysdate,'day') from dual;

4. Select months_between(sysdate,dob) from stud_info;

5. Select next_day (sysdate,'tuesday') from dual;

6. Select greatest ('12-dec-1999','13-mar-2000') from dual;

7. Select round(dob,'month') from stud_info;

8. Select Avg(to_char(sysdate,'yyyy')-to_char(dob,'yyyy'))from  stud_info;

9. Select first_name,to_char(dob,'dd/month/yyyy') from stud_info;

10. Select count(distinct(city)),city from stud_info group by  city;
```

Lab 8

```
1. Select distinct(last_name) from stud_info group by last_name;

2. select student_id,first_name from stud_info where instr
   (zipcode,'00',1,1) <> 0;

3. Select Course Id, Department Id from course where additional_fees =
(select min(additional_fees) from course);

4. Select Course Id, to_char(Additional_fees,$999.99) from   course;

5. Select avg(additional_fees) from course;

6. Select round(additional_fees,1) from course;

7. Select min(addititional_fees), max(additional_fees), Max( additional_fees
)- min(additional_fees) as different from  course;

8. select schedule_id from sche_type where to_char(  starting_time, 'month')
= to_char( add_months(sysdate,1),  'month') and duration < 8;

9. Select schedule_id from sche_type where starting_time=sysdate;

10. Select * from sche_type where to_char(starting_time,'dd') >10   and
to_char(starting_time < 10);

11. Select schedule_id, duration, to_char( months_between(  starting_time,
sysdate),'dd') from sche_type;

12. Select * from sche_type where to_char(starting_time,'month') >
    to_char(sysdate,'month');

13. Select * from sche_type where to_char(starting_time,'ww')=
    to_char(sysdate,'ww');

14. Select class_building, class_room, NVL(TO_CHAR  (seating_capacity), 'NOT
APPLICABLE') as seating_capacity,  location  from class_loc;

15. Select distinct(class_room),instructor_id from class;
```

Lab 9

```
1. Update stud_info set last_name = 'nil' where last_name is  null;

2. Update stud_info set first_name = 'radiant' where telephone is  null And
email is null;

3. Update stud_info set last_name = 'madrasi' where city =   'chennai';

4. Update stud_info set last_name = 'TTT' where state like 'T%';
```

```
5. Update stud_info set last_name = ‘AAA’ where state like ‘%A’;

6. Update stud_info set first_name = ‘XXX’ and dob =sysdate where  first_name
like ‘___a%’;

7. Update stud_info set last_name = ‘youth’ where to_char(   sysdate,
'yyyy')- to_char(dob,'yyyy') < 20;

8. Update stud_info set last_name = ‘birthday’ where dob =   sysdate;

9. Update stud_info set last_name =’senior’ where dob =(select
   Min(dob) from stud_info) and sex='male';

10. Update stud_info set last_name =’jan’ where to_char(dob,   ’month’) =
‘january’;
```

Lab 10

```
1. Delete from stud_info where Last_name is null;

2. Delete from stud_info where telephone is null and email is   null;

3. Delete from stud_info where city =’chennai’;

4. Delete from stud_info where state like ‘T%’;

5. Delete from stud_info where state like ‘%a’;

6. Delete from stud_info where first_name like ‘___a%’;

7. Delete from stud_info where length(first_name) = 10;

8. Delete from stud_info where to_char( sysdate, 'yyyy')-  to_char(dob,'yyyy')
> 20;

9. Delete from stud_info where to_char(dob, ’month’) =’june’ and
   to_char(dob,’dd’) = ‘05’;

10.  Delete from stud_info where to_char(dob, ‘month’)=’october’;

11.  Delete from stud_info where dob =(select Min(dob) from
   stud_info) and sex='male';
```

Lab 11

```
1. Rename stud_info to student;

2. Grant alter, update, insert on student to msit;

3. Revoke msit;

4. savepoint s1;

5. Insert into student_info values ('&Student_id','&Last_name',
   '&First_name','&Dob', '&Address', '&city', '&State',    '&ZipCode',
'&Telephone', '&Fax', '&Email','&sex');

6. Rollback;

7. Commit;

8. Rename student to stud_info;

9. Create table employee as select * from inst;

10. Alter table inst disable inst_pk;

11. Truncate employee;

12. Alter table employee drop dept_fk;

13. alter table employee add(age raw(2) primary key);

14. drop table employee;
```

Lab 12

```
1. Select 'MSU techno wing &'||' CIT@MSU &'||' RADIANT'||' ray of  hope' from
dual;

2. Select concat(first_name,last_name) from inst;

3. Create table employee (Empid varchar2(4),Joindate date, Basic
   number(10,2),HRA number(8,2),DA number(8,2),PF number(8,2));

4. Insert into employee values('&Empid', '&Joindate', &Basic,   &HRA, &DA ,
&PF);

5. Select basic+hra+da-pf as netsalary from employee;

6. Select (3000*3*4)/100 from dual;

7. Select Empid from employee where basic = 3000;

8. select * from employee where basic <> 4000;
```

```
9.  Select * from employee where HRA > 750;

10. Select * from employee where DA < 1000;

11. Select * from employee where pf between 500 and 1000;

12. Select * from employee where Empid in('e101','e103','e104');

13. Select abs(-12.23) from dual;

14. Select FLOOR(15.7) "Floor" FROM DUAL;

15. Select ceil(15.7) "ceiling' from dual;

16. Select (to_char(sysdate,'dd'))-(to_char(joining_date,'dd'))  from employee;
```

Lab 13

```
1.  select first_name from stu_info where student_id =(select  student_id from
stud_sche where grade = 'A');

2.  Select duration from sche_type where schedule_id =(select  schedule_id from
sche_head where schedule_description = 'third   shift';

3.  select additional_fees from course where department_id  =(select
department_id from depat_info where department_name =   'computer';

4.  Select first_name from inst where department_id = 'd03';

5.  Select semester from course where schedule_id = (select   schedule_id from
sche_head where schedule_description ='first  shift';

6.  Select count(*) from stud_info;
7.  select sum(duration) from sche_type where day = 1;

8.  Select avg(basic) from employee;

9.  Select min(basic) as min_basic, max(basic) as max_basic from   employee;

10. Select count (*) from employee where basic >= 2000;

11. Select count (*) from stud_info where telephone is null;

12. Select count (*) from stud_info where state ='TN' and
    to_char(dob,'month')='August';

13. Select concat(first_name,last_name) from stud_info;

14. Select replace (instr(first_name ,'a',0,1),'a','b') from  stud_info;

15. Select upper (concat(first_name,Last_name)) from inst;
```

16. Select lower (schedule_description) from sche_head;

Lab 14

1. Create view student as select student_id,first_name,last_name,
   department_id from stud_info ,depart_info ;

2. Update student set first_name = ‘bhavani’ where first_name = ’saran’;

3. Create synonym cours for course;

4. Create sequence instseq START WITH 0 MINVALUE 1 INCREMENT BY 1 MAXVALUE 20 CYCLE CACHE 10;

5. Alter sequence instseq maxvalue 15;

6. Create index stud on stud_info(first_name);

7. Create index stud1 on stud_info(first_name);

8. Create view course_view as select Course_id,Department_id,
   Title,Description,Additional_fees, (units *250 + additional fees) as Total_fees from course;
9. Create view class_summary as select Class_id, Class_building, Class_room, Department_id, Title, First_name, Last_name from class, instructor where Class.department_id = course.department_ id and Class.course_id = course.course_id and Class.instructor_id = instructor.instructor_id;

10. Create view dummy_view as select name,id from dummy;

11. Create table dummy(name varchar2(20),id number(9));

12. Create view class_view as select * from class;

13. Create view class_summary as select * from class_view;

14. Update class_view set description = ‘shift’ where description is null;

15. Delete from class_view where department_id is null and additional_fees > 100 and total_fees < 500;

16. Create synonym class_alias for class;

17. Rename class_alias to clas;

18. Create synonym course_alias for course;

19. Drop synonym course_alias;

Lab 15

1. Create index stud_last on stud_info(last_name);

2. Create index inst_first on inst(first_name);

3. Create index dept_name on depart_info(department_id);

4. Drop index inst_first;

5. Drop index stud_last;

6. Drop index dept_name;

7. Create cluster dept_cluster(number(2));

8. Create table dept(deptno number(2), dname varchar2(30), loc varchar2(20));
9. Create table emp(Empno number(3),Ename varchar2(30),Doj date,Desig varchar2(10),Sex char(1),Deptno number(2),Salary number(7,2),Comm number(7,2));

10. Create sequence deptno_sequence START WITH 1 INCREMENT BY 1 MAXVALUE 99 CYCLE CACHE 10;

11. Create sequence empno_sequence START WITH 100 INCREMENT BY 5 MAXVALUE 999 CYCLE CACHE 10;

12. Insert into dept(deptno_sequence,'computer','hgvg');

13. Insert into emp(empno_sequence,'saran','12-mar-74', 'technical', 'm', deptno_sequence,12000.00,2000.00);

Lab 16

1. Create type address(Add1 number(5),Add2 varchar2(10),Add3 varchar2(5),Add4 number(7));

2. Create type emp(Eno number(2), Ename varchar2(10),Eadd address);

3. insert into emp values(100,address(100, first st, Chennai, 600078);

4. insert into emp values(101,address(102, iii phases, bgl, 576897);

5. select * from emp where eadd='10';

6. Update emp set eadd = address(10, ii st, mds,35567) where eno = 100;

7. Delete from emp where eadd(address(add1))='10';

8. Create type personal(Name varchar2(20),Age number(3),Sex char(1));

9. Create table person(Id number(4),Centre varchar2(15),Per personal);

10. Insert into person values(1234,'Chennai', personal(meena, 22,f));

    Insert into person values(2345,'nellai', (veena, 23,f));
    Insert into person values(3456,'Bangalore', (reena, 24,f));
11. Drop type personal;

```
12. Create type person_det(Name varchar2(15),Age number(2),Desg  varchar2(15));

13. Create table person_det_table(p person_det);

14. Create table person_store_table(Dept_name varchar2(10), Dept_no
number(3),Person_info person_details_table_ty);

15. Create type price as varray(3) of varchar2(8);

16. Create type dept_type(Deptno number(2), Dname varchar2(14),Loc
    varchar2(13));

17. Create table dep(d dept_type);

18. Create table emp(Empno  number(4),Job varchar2(10),Mgr  number(4),Hiredate
date,Sal number(7,2),Comm number(7,2),Dept    dept_type);
```

Lab 17

```
1.  begin
    if salary > 2500 and salary < 3000 then
     salary = salary + 10 /100;
    end if;
    end;

2. begin
    if salary > 3000 and salary < 5000 then
     salary = salary + 13 / 100;
    end if;
   end;

3. begin
    if salary > 5000 then
      salary = salary + 15 / 100;
   end if;
   end;

   while loops

4. declare
   y Boolean = 'true';
   begin
    while(y = true)
    loop
   insert into stud_info values('&Student_id','&Last_name',
   '&First_name','&Dob', '&Address', '&city', '&State', '&ZipCode',
   '&Telephone', '&Fax', '&Email','&sex');
   dbms_output.put_line('do you want to continue');
   y := &y;
     end loop;
     end;
```

5.
```
declare
 y Boolean = ‘true’;
begin
 while(y = true)
 loop
insert into depart_info values('&department_id',
'&department_name','&hod');
dbms_output.put_line(‘do you want to continue’);
y := &y;
  end loop;
  end;
```

6.
```
declare
s first_name%type;
begin
select first_name into s from emp where salary > 3000;
dbms_output.put_line(s);
end;
```

7.
```
declare
s emp%rowtype;
begin
select * into s from emp where sex =’male’;
   dbms_output.put_line(s);
end;
```

8.
```
declare
t number(9,2);
begin
 select (salary + comm) as total_salary from emp where comm is not null;
end;
```

9.
```
begin
 loop
insert into inst values ('&INSTRUCTOR_ID', '&DEPARTMENT_ID', '&LAST_NAME',
'&FIRST_NAME', '&TELEPHONE', '&FAX','&EMAIL','&SEX','&POSITION');
exit when (INSTRUCTOR_ID %rowcount > 20);
end loop;
end;
```

10.
```
declare
t number(9,2);
begin
loop
 select (salary + comm) as total_salary from emp where comm is not null;
exit when total_salary > 25000;
end loop;
end;
```

11.
```
declare
 s emp.eno%type;
 begin
 dbms_output.put_line(“enter the empno”);
```

```
s := &s;
delete from emp where eno = s;
end;
```

12.
```
declare
  d dept.department_id%type;
  d1 dept.department_name%type;
  begin
select department_id,department_name into d,d1 from depart_info where
mod(department_id,2)<>0;
dbms_output.put_line(d,d1);
end;
```

Lab 18

1.
```
Declare cursor c1 is
Select sql from emp;
S emp.sal % type;
Begin
  Open c1;
  Loop
  Fetch c1 into s;
  Exit when c1%notfound;
  Update emp set sal=sal+s*15/100;
  End loop;
  Close c1;
End;
```

2.
```
declare cursor c2 is
  select additional_fees from course;
  a course additional_fees%type;
  begin
     open c2;
     loop
     fetch c2 into a;
     exit when c2%notfound;
update course set additional_fees = a-a * .05;
  end loop;
  close c2;
  End;
```

3.
```
declare cursor c3 is
  select additional_fees from course;
  a course additional_fees % type;
begin

  open c3;
  loop
  fetch c3 into a;
  exit when c3%notfound;
  if (a+(a*10/100)> 100) then
```

```
      update course set additional_fees = a-(a*20/100);
      else
      update course set additional_fees = a+(a*10/100);

   end if;
   end loop
   end;
```

4.

```
   declare cursor c4 is
     select * from emp;
        j emp.%rowtype;
     begin
        open c4;
        loop
        fetch c4 into j;
        exit when c4%notfound;
        select to_char(doj, 'month') from emp;
        end loop;
        end;
```

5.

```
      declare
        s sche_type.shedule_id % type;
        st sche_type.starting_time % type;
        d  sche_type.duration % type;
        c  class.class_room % type;
      begin
        c := &c;
        select starting_time, duration into st, d from sche_type
   where schedule_id = (select schedule_id from class  where class_room = c);
   dbms_output.put_line(st,d);
   end;
```

6.

```
   declare
        f inst.first_name%type;
        c class.class_room%type;
   begin
        f := &f;
        c := &c;

        select i.instructor_id,i.Department_id,
   i.last_name,c1.day,c1.duration,c1.starting_time,
   c1.schedule_id, c1.class_room from class a, inst i,
   sche_type c1 where inst.first_name = &f and
   class.class_room = &c;
   end;
```

7.

```
   begin
     select schedule_description, s.schedule_id, s.day,
```

```
      s.starting_time, s.duration from sche_head, sche_type;
   end;

8. begin
   create table ins as select instructor_id, first_name, last_name from inst;
   insert into ins values(&instructor_id,&first_name, &last_name);
     end;

9. begin
     delete from ins;
   end;

9. begin
   select c.class_id,c.course_id,c.department_id, i.instructor_id,
   i.first_name from class c, inst I;
   end;
```

Lab 19

```
1. declare
        s sche_type.shedule_id % type;
        st sche_type.starting_time % type;
        d  sche_type.duration % type;
        c  class.class_room % type;
      begin
        c := &c;
           select starting_time, duration into st, d from sche_type
      where schedule_id = (select schedule_id from class  where class_room =
      c);
      dbms_output.put_line(st,d);
      exception
      when no_data_found then
      dbms_output.put_line('no record found');
      end;
2.
      declare
           f inst.first_name%type;
           c class.class_room%type;
      begin
           f := &f;
           c := &c;

           select i.instructor_id,i.Department_id,
      i.last_name,c1.day,c1.duration,c1.starting_time,
      c1.schedule_id, c1.class_room from class a, inst i,
      sche_type c1 where inst.first_name = &f and
      class.class_room = &c;
      exception
      when no_data_found then
      dbms_output.put_line('no record found');
      end;
2.
```

```
        begin
      insert into course values('&course_id','&DEPARTMENT_ID',
      '&TITLE','&DESCRIPTION',&ADDITIONAL_FEES,&UNITS);
      exception
      when dup_val_on_index then
      dbms_output.put_line('duplicate record');
      end;
3.
   begin
     insert into depart_info values('&department_id',
   '&department_name','&hod');
   exception
   when value_error then
   dbms_output.put_line('value error');
   end;
4.
        declare
      ex1 exception;
      n number(5) := &x;
      begin
        if n > 20000 then
         raise ex1;
         end if;
         exception
         when ex1 then
         dbms_output.put_line('value must not exceed 20000');
         end;

5. declare

      err_num number;
      err_msg varchar2(100);
      begin
         insert into temp(empno) values (null);
      exception
         when others then
         err_num := sqlcode;
         err_msg := substr(sqlERRm,1,100);
         insert into errors values(err_num, err_msg);
      end;
```

Subprograms and Packages

```
1. create or replace procedure dele(d in varchar2(200))is

   begin
     delete from class where description = d;
   end;
   exec dele('Information Technology');

2. create or replace procedure disp(s in varchar2(20))is

   begin
   select c.class_id,c.class_building,c.class_room,
   s.student_id,s.first_name,s.dob, i.instructor_id, i.first_name from class
   c,stud_info s,inst i;
        end;

3. create or replace function dis(sid varchar2(20),cid

   varchar2(20)) return varchar2 is
   begin
    select class_id into c from stud_sche where student_id = (select
   student_id from stud_info where student_id = sid);
   if c = cid then
   return ' no conflict';
   else
   return ' conflict';
   endif;
   end;
4. create or replace function inset return varchar2 is

   begin
     insert into class values('&CLASS_ID','&SCHEDULE_ID',
    '&CLASS_BUILDING','&CLASS_ROOM','&COURSE_ID','&DEPARTMENT_ID',
    '&INSTRUCTOR_ID','&SEMESTER','&SCHOOL_YEAR');
     exception
     dbms_output.put_line('duplicate occur');
     return ('inserted');
     end;

5. create or replace function cons return Boolean is
   begin
     select * from class;
   return (0);
   end;
6. Create or replace package over is
   Procedure add(n number,n1 number,n2 number,n3 number,n4 number);
   Procedure sub(n interger,n1 integer);
   procedure mult(n integer,n1 integer,n2 integer);
   end;

   create or replace package body over as
   procedure add(n number,n1 number,n2 number,n3 number,n4 number)is
```

```
begin
n := n+n1+n2+n3+n4;
dbms_output.put_line('the value of '||n);
end;
procedure sub(n interger,n1 integer)is
begin
n := n – n1;
dbms_output.put_line('the value of '||n);
end;
procedure mult(n integer,n1 integer,n2 integer)is
begin
n := n * n1 * n2;
dbms_output.put_line('the value of '||n);
end;
end;
```

7.
```
create or replace package pack1 is
function reg(std varchar2(20),cid varchar2(20)) return Boolean;
function dis(sid varchar2(20),cid varchar2(20)) return varchar2;
procedure ins(cid varchar2(20),ind varchar2(20));
procedure stu(sid varchar2(20),cid varchar2(20),gr varchar2(2));
function avg(s varchar2(20)) return Boolean;
end;

create or replace package body pack1 is
begin
function reg(std varchar2(20),cid varchar2(20))return Boolean is

begin
select class_id into c from stud_sche where student_id = (select student_id
from stud_info where student_id = sid);
if c = cid then
message ' registered';
else
message ' not registered ';
end;

function dis(sid varchar2(20),cid varchar2(20)) return varchar2 is

begin
 select class_id into c from stud_sche where student_id = (select
student_id from stud_info where student_id = sid);
if c = cid then
return ' no conflict';
else
return ' conflict';
endif;
end;

procedure ins(cid varchar2(20),ind varchar2(20))
begin
 select class_id into c from class where instructor_id = ind;
if c = cid then
```

```
message ‘ assigned’;
else
message ‘not assigned’;
end if;
end;

procedure stu(sid varchar2(20),cid varchar2(20),gr varchar2(2));
begin
update stud_sche set grade = gr where student_id = sid and class_id = cid;
end;

function avg(s varchar2(20)) return Boolean is
begin
select avg(grade) into g from stud_sche where student_id = s;
dbms_output.put_line(g); return (‘inserted’);
end;
end;
```

Lab 20

1.
```
create or replace trigger t1 before insert on course for each
row

begin
  if not(to_char(sysdate,’month’)=’sunday’ or
to_char(sysdate,’month’)=’saturday’) then
  dbms_output.put_line(‘inserting operation’);
else
dbms_output.put_line(‘insertion not allowed’);
end if;
end;
```

2.
```
create or replace trigger t2 after update of date on class for  each row

begin
dbms_output.put_line(‘updating operation’);
end;
```

3.
```
create or replace trigger t3 before delete on depart_info for   each row

begin
  dbms_output.put_line(‘do you want to delete’);
end;
```

4.
```
create or replace trigger t4 after delete on inst for each row

begin
  insert into temp values('&INSTRUCTOR_ID', '&DEPARTMENT_ID',
'&LAST_NAME', '&FIRST_NAME','&TELEPHONE','&FAX','&EMAIL','&SEX',
'&POSITION');
end;
```

5. Create or replace trigger t5 before insert/update/delete on stud_info for each row

```
begin
  if (to_char(sysdate,'month')='monday' or
to_char(sysdate,'month')='wednesday' or to_char(sysdate,'month')='sunday')
then
  dbms_output.put_line('trigger operation');
else
dbms_output.put_line('insertion not allowed');
end if;
end;
```

6. Create or replace trigger t6 before insert/update/delete on inst for each row

```
Begin
  If not(to_char(sysdate,'hh')>='6 p.m' or to_char(sysdate,'hh')<='10
p.m')  then
  dbms_output.put_line('trigger operation');
else
dbms_output.put_line('insertion not allowed');
end if;
end;
```

Lab 21

1.

```
import java.sql.*;
import java.io.*;
import oracle.jdbc.driver.*;

public class ex6
{
   public static float test (int en, int ri) throws SQLException
   {
     Connection conn = new OracleDriver ( ). DefaultConnection( );
     CallableStatement cstmt = conn.prepareCall("{?=CALL
compute (?,?) }" ) ;

cstmt.registerOutParameter (1.Types.FLOAT);
cstmt.setInt(2,en);
cstmt.setInt(3,ri);
cstmt.executeUpdate( );

int ot = cstmt.getInt(1);

return ot;
 }
}

create or replace function compute (eno number, rise number) return number
as
```

```
rt number;
s1 number;
begin
select sal into s1 from emp where empno = eno;
rt := s1 + s1 * (rise/100)return rt;
end;
```

2.

```
import java.sql.*;
import java.io.*;
import oracle.jdbc.driver.*;

public class ex3
{
  public static int esal(int eno)throws SQLException
  {
     Connection conn = new OracleDriver().defaultConnection();
     String sql = "Select Sal From Emp Where Empno= ?";
     Int s1 = 0;

     preparedStatement psmt = conn.prepareStatement (sql);
     psmt.setInt (1,eno);
     ResultSet rset = psmt.executeQuery();

     While (rset.next())
     {
        s1 = rset.getInt (1);
     }
     rset.close();
     psmt.close();

     return s1;
  }
}
```

3.

```
import java.sql.SQLException;
import oracle.sqlj.runtime.Oracle;

#sql iterator MyIter1 (String ENAME, int SAL);

class Ts1
{
  public static void main (String args[]) throws SQLException
  {
     try {
     oracle.connect (Ts1.class, "connect.properties");
     Ts1 ti = new Ts1();
     Ti.runExample ();
```

```
        } catch (SQLException e) {
        System.err.println ("Error running the example: here only" + e);
    }
} // End of method main

// Method that runs the example
void runExample() throws SQLException
{
  //Issue SQL command to clear the SALES table

     int dp = 20;
   MyIterl iter;
   #sql  iter = { SELECT ENAME, SAL FROM EMP WHERE DEPTNO = :dp };

   while (iter.next())   {
  System.out.println(iter.ENAME() + "   " + iter.SAL());

  }
   }
}
```

Lab 22

1.

```
import java.sql.SQLException;
import Oracle.sqlj.runtime.Oracle;

class Ts2

{
public static void main (string args[]) throws SQLException
  {
     try {
        Oracle.connect(Ts2.class, "connect.properties");
        Ts2 ti = new Ts2();
        ti.runExample ();
     } catch (SQLExeception e) {
System.err.println("Error running the example: here only " + e);
     }
  } // End of method main

//Method that runs the example void runExample() throws SQLException
{
   //Issue SQL command to clear the SALES table
    #sql {
  BEGIN
FOR I IN 1..10 LOOP
  INSERT INTO SALES VALUES('SLS'//I);
END LOOP;
END;
 };
#sql {COMMIT};
}
```

```
    }

2.
    import java.sql.SQLException;
    import oracle.sqlj.runtime.Oracle;

    #sql iterator MyIter (String ITEM_NAME);

    class Ts
    {
      public static void main (String args[]) throws SQLException
      {
      try {
         Oracle.connect (Ts.class, "connect.properties");
         Ts ti = new Ts();
         ti.runExample ();
          }catch (SQLException e)  {
      System.err.println ("Error running the example: here only" + e);
    }
    } //End of method main

    //Method that runs the example
    void runExample() throws SQLException

    {
    // Issue SQL command to clear the SALES table

    #sql { DELETE FROM SALES };
    #sql { INSERT INTO SALES (ITEM_NAME) VALUES ('Hello, SQLJ!') } ;
    #sql { INSERT INTO SALES (ITEM_NAME) VALUES ('Hello, SQLJ!!!!!!')};
    #sql { COMMIT } ;
    MyIter iter;
    #sql iter = { SELECT ITEM_NAME FROM SALES };

    while (iter.next())   {
      System.out.println(iter.ITEM_NAME());
    }
    }

    }

3.
    import java.sql.SQLException;
    import Oracle.sqlj.runtime.Oracle;

    #sql iterator Empiter(int,String, Float);

    class Ts3

    {
     public static void main (string args[]) throws SQLException
      {
      try {
```

```
Oracle.connect(Ts3.class, “connect.properties”);
Ts3 ti = new Ts3();
ti.runExample ();
   } catch (SQLExeception e) {
 System.err.println(“Error running the example: here only “ + e);
   }
   } // End of method main

//Method that runs the example
void runExample() throws SQLException
{
  int eno = 0;
  String en = null;
  float sl = 0.0f;
  Empiter mp;
  # sql mp = { SELECT EMPNO,ENAME,SAL FROM EMP };

while(true)
   {
   #sql {  FETCH  : mp INTO :eno, :en, :sl};
   if(mp.endFetch())break;
   System.out.println(eno);
   System.out.println(en);
   System.out.println(sl);
      }
    }
  }
```

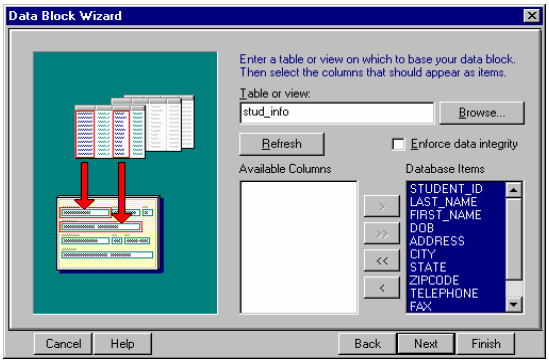
Lab 23

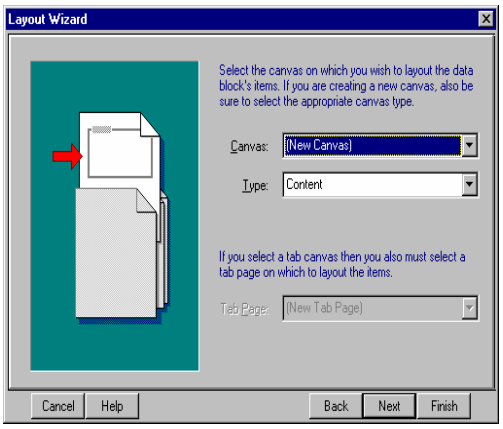
1. Create a new form module for student information table. Create a new block by using the data block wizard.

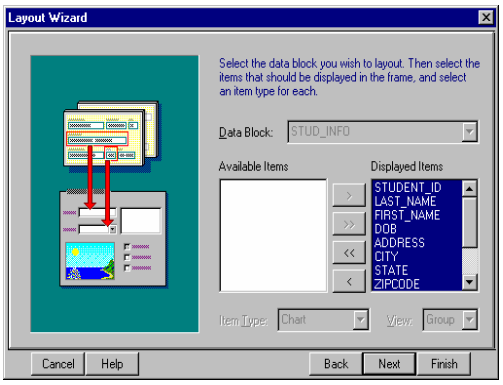
   Choose all the fields from the stud_info table and select all fields.

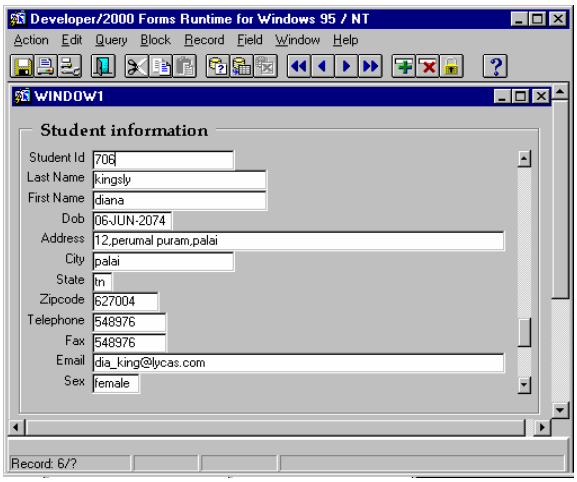
   Choose the layout wizard and select the type of canvas from this.

   Choose all fields from the stud_info table and select all fields.









2. No formal solution. Same above solution are continuing for this question.

3. No formal solution.

Lab 24

1. No formal solution.

2. No formal solution.

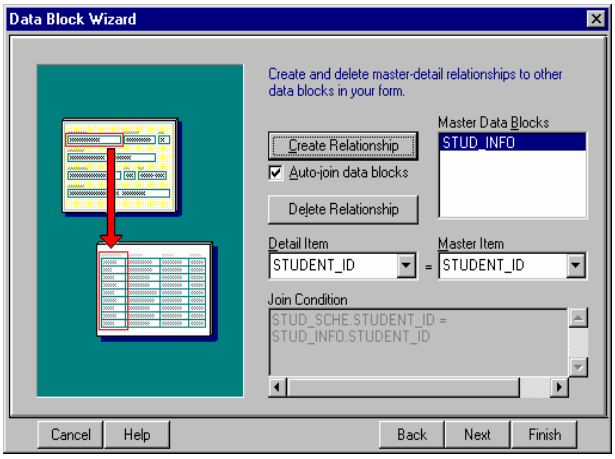
3. No formal solution.

4. No formal solution.

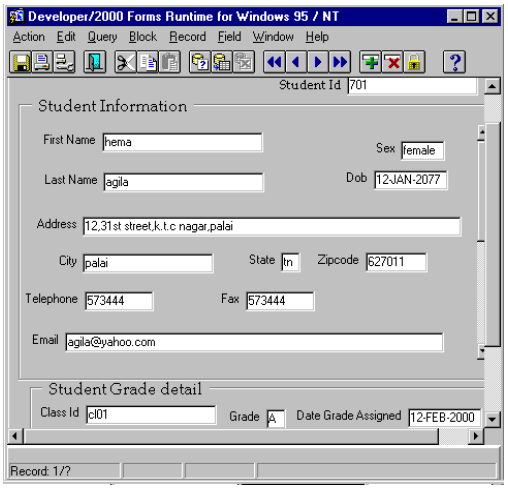
5. No formal solution.

Lab 25

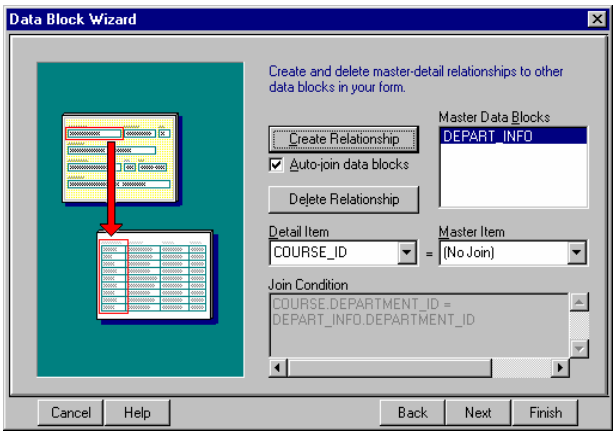
1.



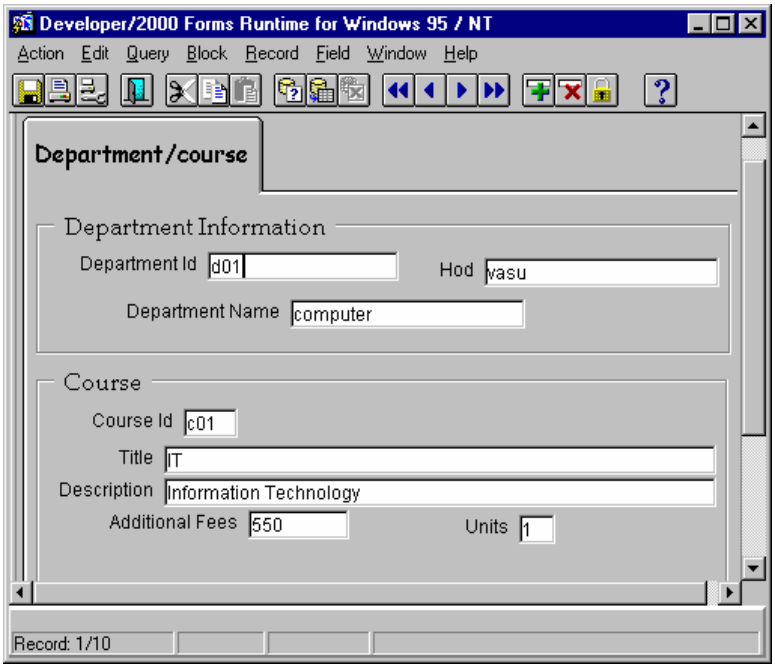
create the relation for both stud_info and stud_sche table using student_id and run the form.



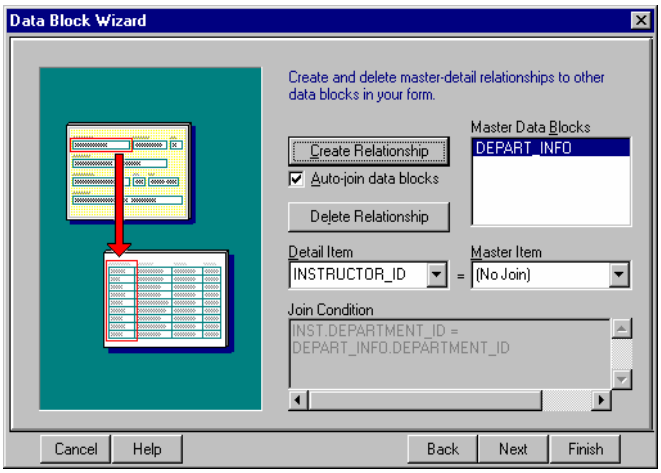
2.

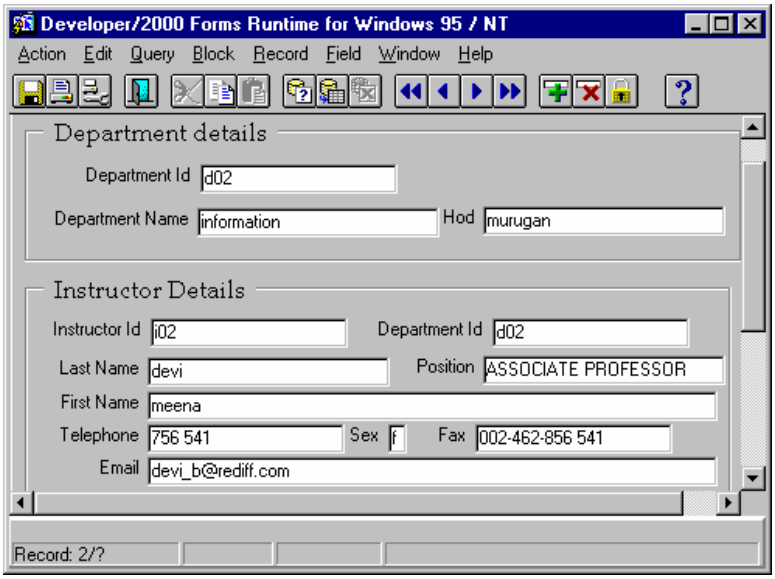


Create the relationship for department and course table and join these two tables.

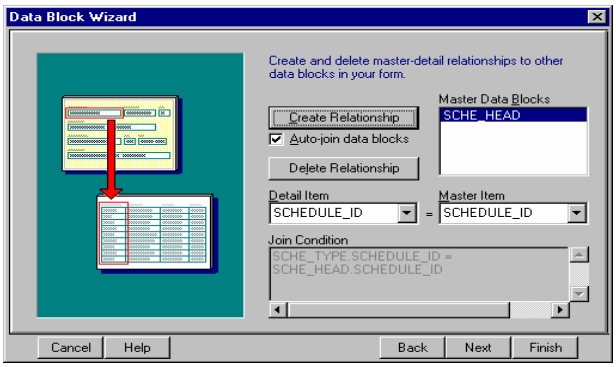


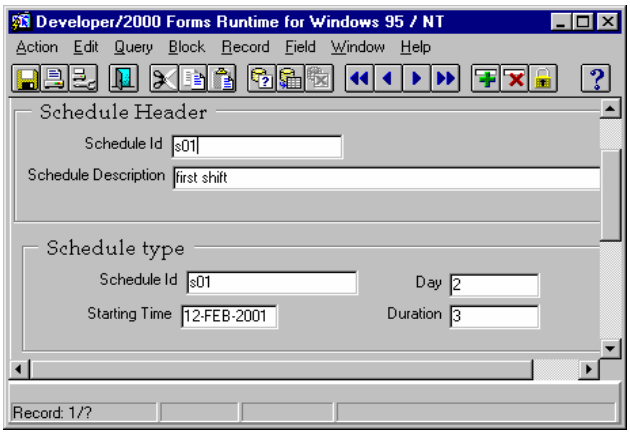
3. create the relation for department and inst table using depatment_id as key.


Data Block Wizard
Create and delete master-detail relationships to other data blocks in your form.
Master Data Blocks
Create Relationship
DEPART_INFO
Auto-join data blocks
Delete Relationship
Detail Item
Master Item
INSTRUCTOR_ID
=
(No Join)
Join Condition
INST.DEPARTMENT_ID =
DEPART_INFO.DEPARTMENT_ID
Cancel
Help
Back
Next
Finish


Developer/2000 Forms Runtime for Windows 95 / NT
Action Edit Query Block Record Field Window Help
Department details
Department Id d02
Department Name information
Hod murugan
Instructor Details
Instructor Id i02
Department Id d02
Last Name devi
Position ASSOCIATE PROFESSOR
First Name meena
Telephone 756 541
Sex f
Fax 002-462-856 541
Email devi_b@rediff.com
Record: 2/?

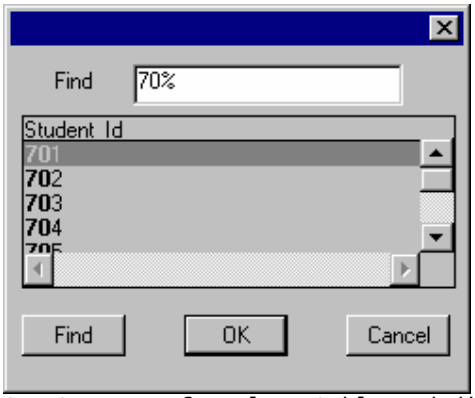
4. create the relation for sche_head and sche_type using schedule_id.



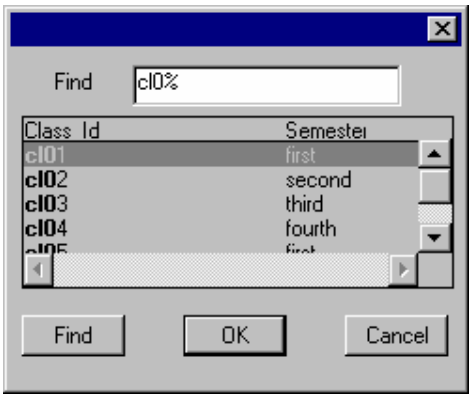


Lab 26

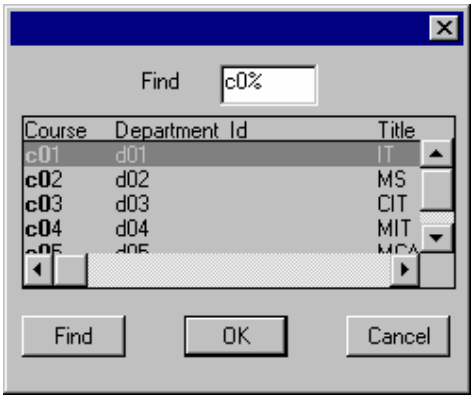
1. create LOV for student_id and display the student_id for using LOV. and execute the form.



2. Create a Lov for class table and display the class room detail.



3. create a lov for course table and display the course detail.



Lab 27

1. Push Button: Exit
   Trigger : When_Button_Pressed
   PL/SQL Code
   Exit_form;

   Item type: Radio Button
   Trigger : When_Radio_Changed
   PL/SQL code

```
declare
  ss number (8,2);
na char (10);
do date;
begin
  select ename, doj, salary into na, do, ss from bonus
  where eno = :eno;
     :ename := na;
  :doj   := do;
     :salary := ss;

if  :radio_group4 = 1 then
```

```
:bonus := ss * .2;

elsif :radio_group4 = 2 then
  :bonus := ss * .9;

elsif :radio_group4 = 3 then
  :bonus := ss * .8;

elsif :radio_group4 = 5 then
  :bonus := ss * .6;

endif;
end;
```

2. Push button : calculate

```
Trigger    : When_Button_Pressed
PL/SQL Code
Begin
  :tocharge := :loccall + :stdchr;
  End;

Push button : Exit
Trigger    : When_Button_Pressed
PL/SQL Code
Exit_form;
```

Lab 28

1)

```
Push_button  : view
Trigger : When_button_pressed

PL/SQL editor

Declare
  al boolean;
Begin
al:= show_lov('lov12');
End;

Push button  : find
Trigger    : When_button_pressed

PL/SQL editor

Declare
pn char(10);
qu number(7);
up number(8,2);
rl number (8,2);
```

```
Begin
 Select pname, unitprice, quantity, reorderd into pn,up,qu,rl from Stock
where pid = :pid;
:pname := pn;
:unitprice := up;
:quantity  := qu;
:reorderd  := rl;

end;

Push_button : Exit
Trigger   : When_button_pressed

PL/SQL editor

 exit_form;

Push_button : previous
Trigger   : when_button_pressed

PL/SQL editor

 previous_record;

Push_button : Last
Trigger     : When_button_pressed

PL/SQL editor
 last_record;

Push_button  : Next
Trigger      : When_button_pressed

PL/SQL editor

 next_record;

Push_button : first
Trigger     : When_button_pressed

PL/SQL editor

 go_block('sto');
 first_record;
 execute_query;

Push_button : Delete
Trigger     : When_button_pressed

PL/SQL editor

 delete_record;
```

```
commit_form;

Push_button : edit
Trigger     : When_button_pressed

PL/SQL editor

go_item('pid');
execute_query;
commit_form;

Push_button : Save
Trigger     : When_button_pressed

PL/SQL editor

commit_form;

Push_button : clear
Trigger     : When_button_pressed

PL/SQL editor
go_block('sto');
clear_block(no_commit);
go_item('pid');
```

Sales

```
Declare
al number;
Begin
go_block('sto');
go_item('pid');
If (:quantity - :ordqty) <= :reorderd then
   al := show_alert('alert 35');
Else
Insert into stock values(:pid, :pname, :unitprice, :quantity, :reorderd,
:costprice, :sellingprice, :ordqty);
   al :=show_alert('alert 36');
   :quantity := :quantity - :ordqty;
   :costprice := :unitprice * :quantity;
   :sellingprice := :costprice * 5;
   execute_query;
   commit_form;
end if;
end;
```

2)

```
Text_box Age
Trigger key  : next_item

If :age <0 or :age >120 then message ('Invalid Age');
```

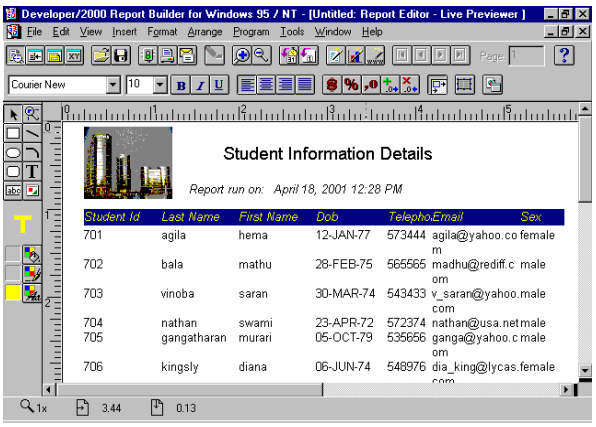
```
go_item('Age');
End if;

Text_box : pcat
Trigger key : Next_item

If ;pcat = 'IP' or :pcat = 'op' then
 If :pcat = 'op' then
 :rent := 0;
 :op_fees :=0;
 set_item_property ('rent',enabled,property_off);
 set_item_property ('op_fees', enabled, property_off);
 go_item('Disease');
end if;
else
 message('Invalid patient category');
 :pcat ='';
 go_item('pcat');
 end if;

Text_box  : Disease
Trigger : key_next_item

PL/SQL code

:Tcharge :=0;
go_item ('cfees');

Text_box   : Inject fees
Trigger : key_next_item

If :pcat = 'IP' then
 go_item('medcharge');
Else
 go_item('op_fees');
End if;

Button  :Add
trigger :When_button_pressed

clear_block (no_commit);
go_back('pat');
go_item('pid');

Button  :  Save
Trigger : When_button_pressed

clear_block (do_commit);
Set_item_property ('op_fees',enabled,property_on);
Set_item_property ('rent',enabled,property_on);
Set_item_property ('inpatient',enabled,property_on);
Set_item_property ('outpatient', enabled,property_on);
```

```
Button  : Select
Trigger :When_button_pressed

Declare
 F number;
Begin
Select pid, age, pcat, disease, tcharge, cfees, injectfees, op_fees, rent,
medcharge, ecgfees into :pid, :age, :pcat, :Disease, :Tcharge, :cfees,
:injectfees, :op_fees, :rent, :medcharge, ; ecgfees from Host where pid =
:pid;

 Set_item_property('outpatient',enabled,property_on);
 Set_item_property ('inpatient',enabled,property_on);

If :pcat = 'ip' then

 Set_item_property('outpatient',enabled,property_off);

Else
 Set_item_property('inpatient',enabled,property_off);
End if;
Exception
 When no_data_found then
 F := show_alert('error');
End;

Button  : outpatient
Trigger : When_mouse_click

Begin

If :pcat = 'op' then
:Tcharge := :cfees + :injectfees + :op_fees + :rent + :medcharge +
:ecgfees;

update Host set Tcharge = :Tcharge where pid = :pid;

 set_item_property ('Inpatient',enabled,property_on);

else
 message('invalid usage');
end if;

exception

 When no_data_found then message('no data found');
End;

Inpatient
 When_mouse_click

Declare
```

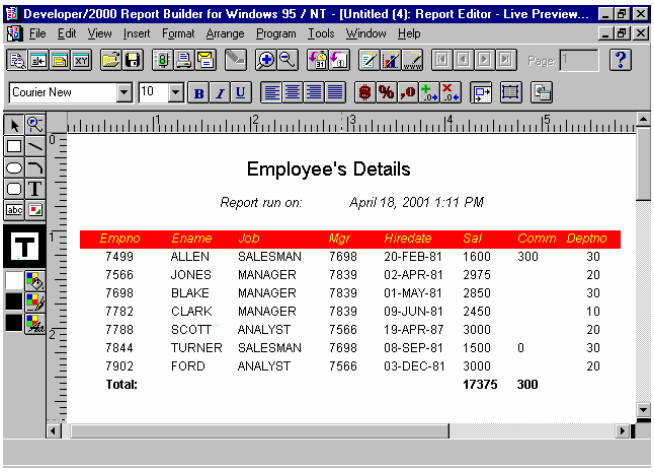
```
T1 number(8,2);
T2 number(8,2);
T3 number(8,2);
T4 number(8,2);
T5 number(8,2);

Begin

If :pcat = 'IP' then
Select cfees, op_fees,rent,medcharge,ecgfees into T1,T2,T3,T4,T5 from Host
where pid = :pid;

:Tcharge :=T1+T2+T3+T4+T5;
update Host set Tcharge = :Tcharge where pid = :pid;
set_itm_property('Inpatient', enabled,property_on);
else
message('invalid');
end if:

exception
when no_data_found then message ('no data found');
end;

Exit
When_button_pressed
Exit_form;
```

Lab 29

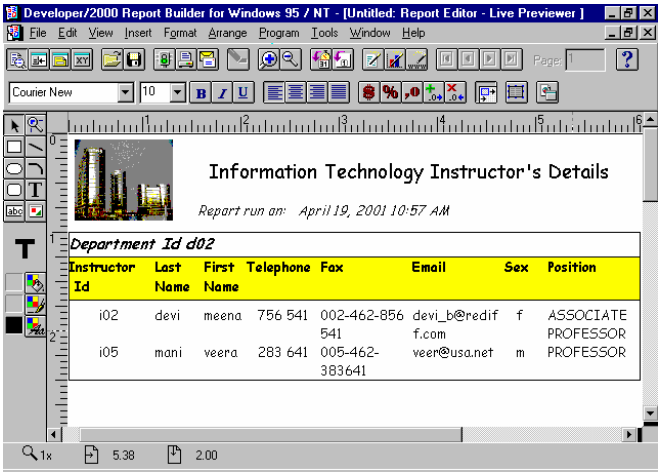
1. create the student information report for stud_info table in tabular form.



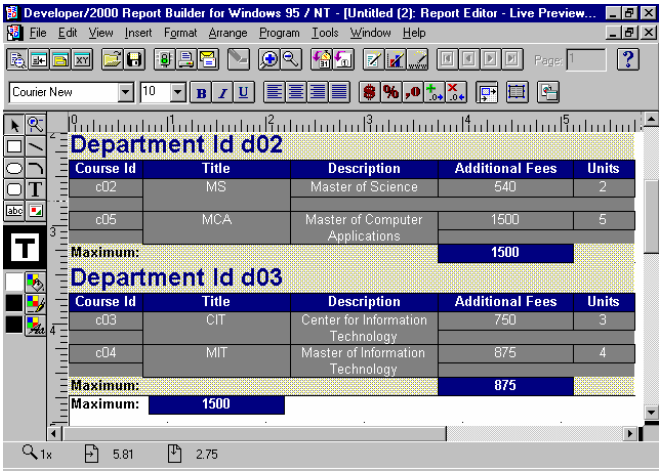
2. create report for employee's detail using report wizard.



3.create report for information technology instructor details.

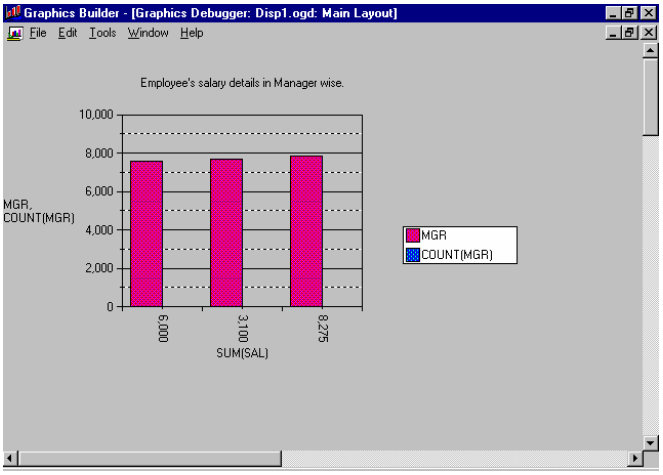


4. Create report for group the course table in department wise and also display the maximum fees.



Lab 30

1. Create the graph for employee's salary details.



2. Create the graph for course details using graphics builder.