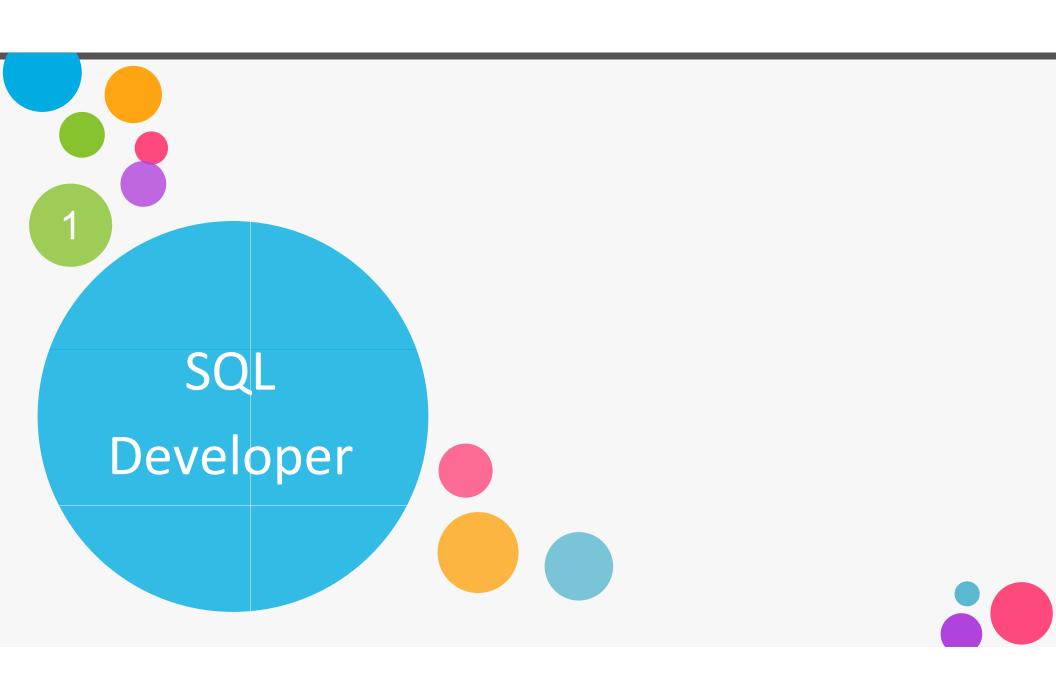




SQL Developer

We are giving a small touch of SQL developer in this class to create DDL scripts for our scott schema tables as we are going to Modify them

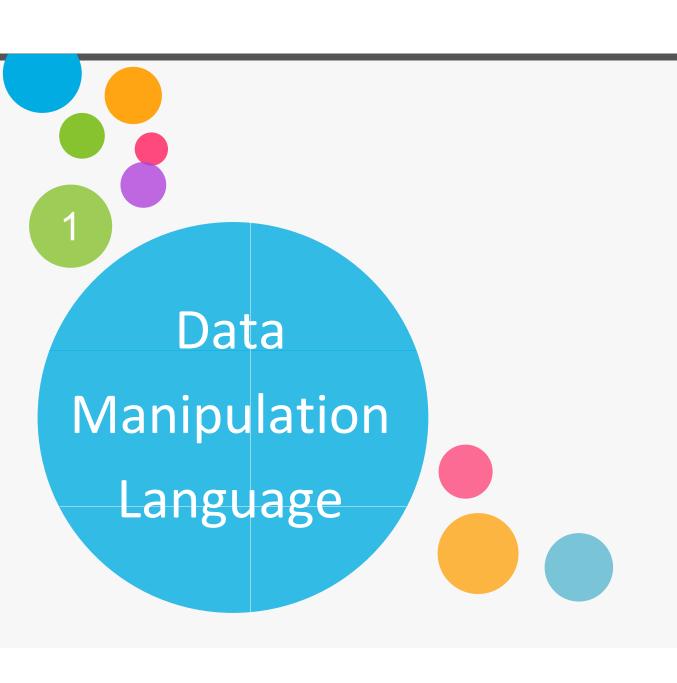




SQL Developer

In class we have seen demo of SQL Developer basics, how to create DDL scripts and data scripts by using SQL Developer (any tool can be used)







Data Manipulation Language

When we want to add, update or delete data in the database, we execute a DML statement.

Transaction

A collection of DML statements that form a logical unit of work.

If you are a bank customer having 2 accounts at a time let say in account 1 or account 2 if you deposit money in account 2 from account 1 then account 1 will show decrease amount transaction, account 2 will show increase amount transaction and there might be a transaction log for a user which will show you log of both transactions.

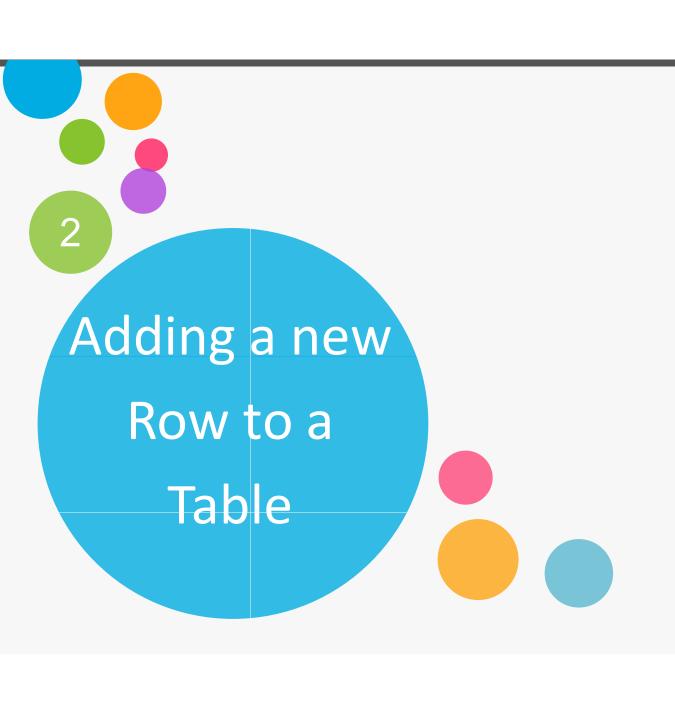


Data Manipulation Language

SQL DML Statements

Notation	Description	
INSERT	Enter new rows into tables	
UPDATE	To change existing rows	
DELETE	To delete existing rows	







Adding a new Row to a table

INSERT Statement is used for adding a new row into a table

Syntax

```
INSERT INTO table [(column [, column ...] ) ]
VALUES (value [, value ...]);
```

Example 1: Insert a new Row in department table as management decide to start a new department in *Detroit(location)* named as *DEVELOPMENT dept.* # allocated to that department is 50.

INSERT INTO dept (deptno, dname, loc) VALUES (50, 'DEVELOPMENT', 'DETROIT');



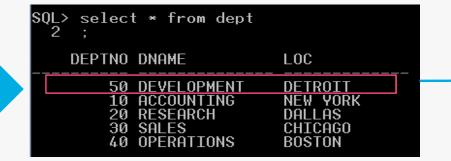
```
SQL> ed
Wrote file afiedt.buf

1 INSERT INTO dept (deptno, dname, loc)

2* VALUES (50, 'DEVELOPMENT', 'DETROIT')

SQL> /

1 row created.
```



Newly entered Record

Note If the column list is not mentioned than all columns must be listed in the default order as they are present in the table(see via desc table command)

So insert statement can be written as:

INSERT INTO dept VALUES (50, 'DEVELOPMENT', 'DETROIT');



Inserting rows with Null values

<u>with column names</u> via mentioning Null values for respective Null Columns this is called <u>Explicit Method</u>

INSERT INTO dept VALUES (70, 'FINANCE', NULL); ←

OR

INSERT INTO dept (deptno, dname, loc) VALUES (70, 'FINANCE', NULL);

Or we Implicitly Omit the columns (to whom we want to set Null) after only Entering <u>all columns with column names</u> this is called Implicit Method

INSERT INTO dept (deptno, dname) – VALUES (60, 'MIS');

→ Location will set to Null

Enforcement of All data types, data ranges and integrity constraints is automatically applied in Oracle so users cant violate them on insertion

Example 2: today company hired an employee Mr *Green* as *SALESMAN* with empno *7196* in Accounting department (deptno *10*) whose Manager is Mr. Clark (empid is *7782*).Company decided to give him salary *2000* per month with *no commission*.

INSERT INTO emp (empno, ename, job, mgr, hiredate, sal, comm, deptno) VALUES (7196, 'GREEN', 'SALESMAN', 7782, SYSDATE, 2000, NULL, 10)



Date function can also been used for inserting specific date value

Example 3: On **03**rd **Feb 1997** company hired an employee Mr *Shaam as Analyst* with empno *2296* in Accounting department (deptno *10*) whose Manager is Mr. Clark (empid is *7782*).Company decided to give him salary **3000** per month with *no commission*.

INSERT INTO emp (empno, ename, job, mgr, hiredate, sal, comm, deptno) VALUES (2296, 'SHAAM', 'ANALYST', 7782, TO_DATE('FEB 3, 97', 'MON DD, YY'), 3000, NULL, 10)

Let see all records of the Employee table



SQL>	select	t * from emp	р;					
	EMPN0	ENAME	J0B	MGR	HIREDATE	SAL	COMM	DEPTNO
	7499	SHAAM SMITH ALLEN	SALESMAN ANALYST CLERK SALESMAN	7782 7902 7698	27-AUG-16 03-FEB-97 17-DEC-80 20-FEB-81	2000 3000 800 1600	300	10 10 20 30
	7566 7654 7698	WARD JONES MARTIN BLAKE CLARK	SALESMAN MANAGER SALESMAN MANAGER MANAGER	7839 7698 7839	22-FEB-81 02-APR-81 28-SEP-81 01-MAY-81 09-JUN-81	1250 2975 1250 2850 2450	500 1400	30 20 30 30 10
	7788 7839 7844 7876 7900	SCOTT KING TURNER ADAMS JAMES FORD	ANALYST PRESIDENT SALESMAN CLERK CLERK ANALYST	7566 7698 7788 7698	19-APR-87 17-NOV-81	3000 5000 1500 1100 950 3000	0	20 10 30 20 30 20
		MILLER	CLERK		23-JAN-82	1300		10

Hiredate saved properly.

Adding a new Row Using substitution variable

INSERT INTO dept (deptno, dname, loc)

VALUES (&department_id, '&department_name', '&location');





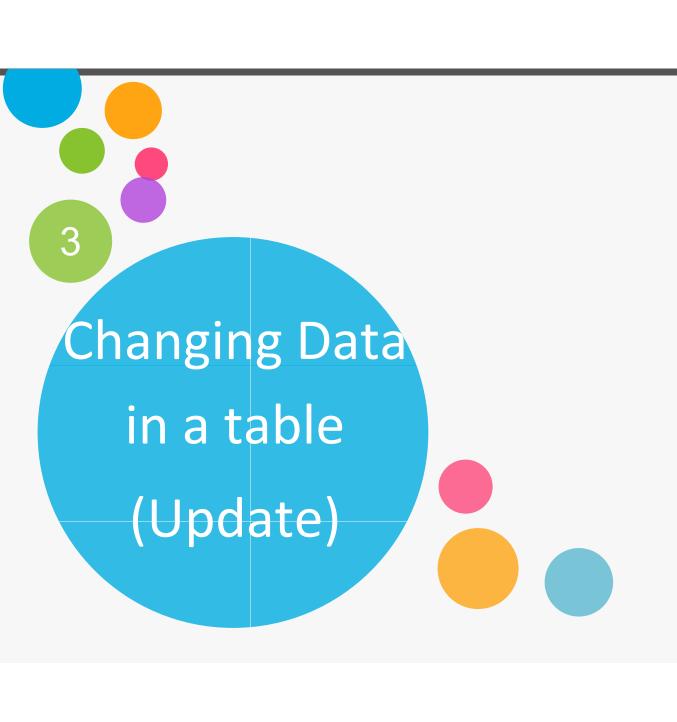
```
1 INSERT INTO dept (deptno, dname, loc)
2* VALUES (&department_id, '&department_name', '&location')
SQL> /
Enter value for department_id: 80
Enter value for department_name: EDUCATION
Enter value for location: ATLANTA
old 2: VALUES (&department_id, '&department_name', '&location')
new 2: VALUES (80, 'EDUCATION', 'ATLANTA')

1 row created.
```

Copy one row of one table to another (same table structures)

To implement this scenario lets create a **LoyalEmp** table as we have already created DDL of emp table lets create this new table using that DDL script and assume that a list of loyal employees are maintained by a company in a that new table.







Changing data in a table (update)

Update Statement is used for changing data in a table

```
UPDATE table

SET column = value [, column = value , ...]

[WHERE condition];

This shows that multiple column values can be updated at a same time

This shows that multiple Rows can Be
```

Example 3: an employee Mr. Clarke recently shifted to a Research department (dept no =20) from accounting. His record needs to be updated. UPDATE emp

SET deptno = 20 WHERE empno = 7782



updated depending on the condition

Changing data in a table (update)

If no where clause is mention in a update query then it will update all rows of the table. If I run the above query in **Loyalemp** table

EMPNO ENAME	J0B	MGR HIREDATE	SAL	СОММ	DEPTNO	_ ₩
7196 GREEN 2296 SHAAM 7782 CLARK 7839 KING 7934 MILLER	SALESMAN ANALYST MANAGER PRESIDENT CLERK	7782 27-AUG-16 7782 03-FEB-97 7839 09-JUN-81 17-NOV-81 7782 23-JAN-82	2000 3000 2450 5000 1300		10 10 10 10 10	efore lpdate

UPDATE Loyalemp SET deptno = 20;

EMPNO ENAME	JOB	MGR HIREDATE	SAL	COMM	DEPTNO	~ >
7196 GREEN 2296 SHAAM 7782 CLARK 7839 KING 7934 MILLER	SALESMAN ANALYST MANAGER PRESIDENT CLERK	7782 27-AUG-16 7782 03-FEB-97 7839 09-JUN-81 17-NOV-81 7782 23-JAN-82	2000 3000 2450 5000 1300		20 20 20 20 20 20	fter Jpdate



Changing data in a table (update)

Updating with Multiple column subquery

Example 3: lets experiment in our Loyalemp table that Mr Miller (7934) JOB, Deptno &

MGR needs to be replace with the Emp's table Mr Blakes (7698) JOB, Deptno & MGR

```
UPDATE loyalemp
```

SET (job, deptno, MGR) =

(SELECT job, deptno, MGR

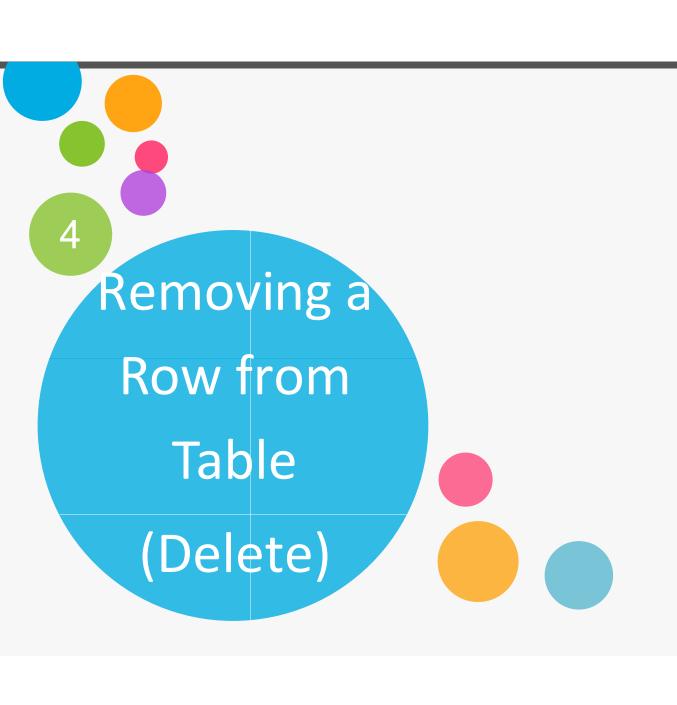
FROM emp

WHERE empno = 7698)

WHERE empno = 7934;

SQL> select * from	ı loyalemp;				
EMPNO ENAME	J0B	MGR HIREDATE	SAL	COMM	DEPTN0
7196 GREEN 2296 SHAAM 7782 CLARK 7839 KING 7934 MILLER	SALESMAN ANALYST MANAGER PRESIDENT MANAGER	7782 27-AUG-16 7782 03-FEB-97 7839 09-JUN-81 17-NOV-81 7839 23-JAN-82	2000 3000 2450 5000 1300		20 20 20 20 20 20







Removing Row from Table (Delete)

Delete Statement is used for Removing existing row from a table **Syntax**

Delete From table [WHERE condition];

Example 1: Remove the **development** department from Departments table

DELETE FROM dept
WHERE dname = 'DEVELOPMENT';

<pre>select * from dept;</pre>	
DEPTNO DNAME	LOC
80 EDUCATION 10 ACCOUNTING 20 RESEARCH 30 SALES 40 OPERATIONS	ATLANTA NEW YORK DALLAS CHICAGO BOSTON



Removing Row from Table (Delete)

All rows of the table will be removed if we will omit where clause

Example 2: Remove all rows from loyalemp table DELETE FROM loyalemp;

```
SQL> select * from loyalemp;
no rows selected
```

Example 3: copy all employees into loyal employees table then remove all those employees who were hired after January 1,1977.

Step 1:

INSERT INTO loyalemp

Select * from emp

Step 2:

DELETE FROM loyalemp

WHERE hiredate > TO_DATE('01.01.97', 'DD.MM.YY');



Removing Row from Table (Delete)

Rot	fore	De	L ati	ion
DC			ICL	

EMPNO ENAME	JOB	MGR	HIREDATE	SAL	COMM	DEPTNO
7196 GREEN	SALESMAN	7782	27-AUG-16	2000		10
2296 SHAAM	ANALYST	7782	03-FEB-97	3000		10
7369 SMITH	CLERK	7902	17-DEC-80	800		20
7499 ALLEN	SALESMAN	7698	20-FEB-81	1600	300	30
7521 WARD	SALESMAN	7698	22-FEB-81	1250	500	30
7566 JONES	MANAGER	7839	02-APR-81	2975		20
7654 MARTIN	SALESMAN	7698	28-SEP-81	1250	1400	30
7698 BLAKE	MANAGER	7839	01-MAY-81	2850		30
7782 CLARK	MANAGER	7839	09-JUN-81	2450		10
7788 SCOTT	ANALYST	7566	19-APR-87	3000		20
7839 KING	PRES I DENT		17-NOV-81	5000		10
7844 TURNER	SALESMAN	7698	08-SEP-81	1500	Ø	30
7876 ADAMS	CLERK	7788	23-MAY-87	1100		20
7900 JAMES	CLERK	7698	03-DEC-81	950		30
7902 FORD	ANALYST	7566	03-DEC-81	3000		20
7934 MILLER	CLERK	7782	23-JAN-82	1300		10

After Deletion

EMPNO ENAME	JOB	MGR	HIREDATE	SAL	COMM	DEPTNO
7196 GREEN	SALESMAN		27-AUG-16			10
7369 SMITH	CLERK	7902	17-DEC-80	800		20
7499 ALLEN	SALESMAN	7698	20-FEB-81	1600	300	30
7521 WARD	SALESMAN	7698	22-FEB-81	1250	500	30
7566 JONES	MANAGER	7839	02-APR-81	2975		20
7654 MARTIN	SALESMAN	7698	28-SEP-81	1250	1400	30
7698 BLAKE	MANAGER	7839	01-MAY-81	2850		30
7782 CLARK	MANAGER	7839	09-JUN-81	2450		10
7788 SCOTT	ANALYST	7566	19-APR-87	3000		20
7839 KING	PRES I DENT		17-NOU-81	5000		10
7844 TURNER	SALESMAN	7698	08-SEP-81	1500	Ø	30
7876 ADAMS	CLERK		23-MAY-87	1100		20
7900 JAMES	CLERK	7698	03-DEC-81	950		30
7902 FORD	ANALYST	7566	03-DEC-81	3000		20
7934 MILLER	CLERK	7782	23-JAN-82	1300		10

Mr. shaam record has been removed

Delete Rows via subquery

Example 3: Delete all loyal employees of sales department

DELETE from loyalemp

WHERE deptno =

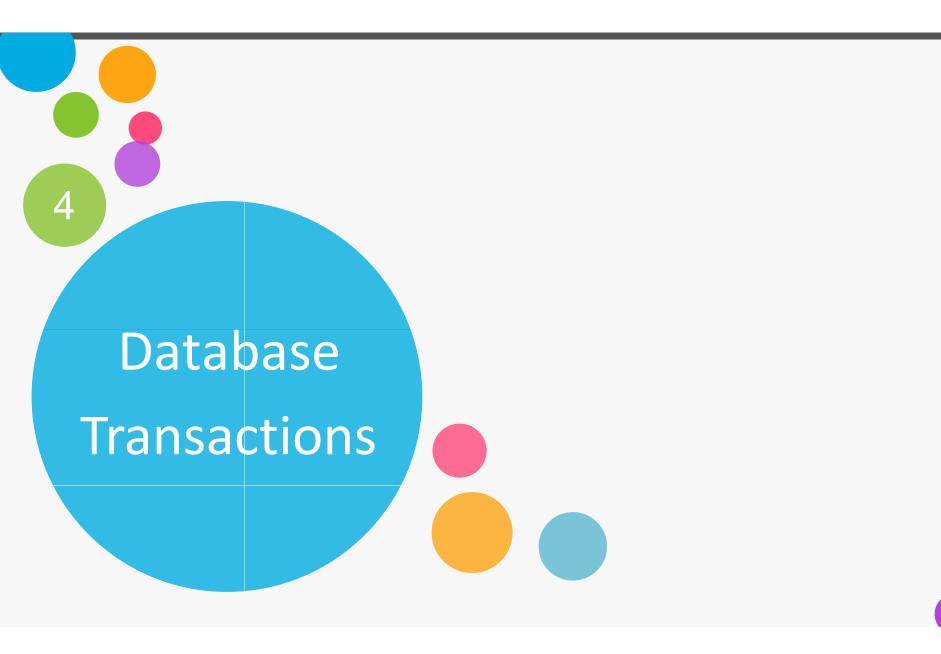
(SELECT deptno

FROM dept

WHERE dname = 'SALES');

	2	DELETE from loyalemp WHERE deptno = (SELECT deptno FROM dept
		WHERE dname = 'SALES')
	6	
5	rov	vs deleted.

EMPNO	ENAME	JOB	MGR	HIREDATE	SAL	COMM		DEPTNO
7369 7566 7782 7788 7839 7876 7902	GREEN SMITH JONES CLARK SCOTT KING ADAMS FORD MILLER	SALESMAN CLERK MANAGER MANAGER ANALYST PRESIDENT CLERK ANALYST CLERK	7902 7839 7839 7566 7788 7566	27-AUG-16 17-DEC-80 02-APR-81 09-JUN-81 19-APR-87 17-NOU-81 23-MAY-87 03-DEC-81 23-JAN-82	2000 800 2975 2450 3000 5000 1100 3000			10 20 20 10 20 10 20 20 10
LOM2 2CT	5666u.						_	





Concept of Database transaction is to maintain consistency that is based on multiple transactions at a time with in a single operation. We know that transaction consist of DML statements that will finally impact one consistent change to the whole data. To maintain that consistency we must be sure that either all of the transactions successfully done(commit) or if few are done or few are not due to some technical issues then transaction must be Rollback to run the whole process again.

Example: funds transfer between two accounts includes debit from one account and credit to another account. Although they are 2 different queries but they both **either run successfully** or if due to some issue debit query successfully run but credit query was unable to run then either of the query will not impact to the tables to maintain consistency <u>unless commit command is executed.</u> This scenario will be <u>Rollback.</u>

Transaction Types

Notation	Description				
Data Manipulation	Consists of any number of DML statements that the Oracle				
Language(DML)	Server treats as a single entity or a logical unit of work after				
	commit.				
Data Definition					
Language(DDL)	Consists of only one DDL statement				
Data Control Language(DCL)	Consists of only one DCL statement				
DCL (data Control languag	ge) is used to control privileges in Database. Let				
say to allow a user to crea	ate table we write following DCL command:				
grant create table to username;					
Similarly to take create table rights we write:					
Revoke create table from	username;				

A transaction begin when the first SQL statement is encountered and then transaction terminates when:

- 1. commit or Rollback statement is issued
- 2. User exits **SQL Plus** or **any** database session
- 3. Machine fails or a system crashes

Otherwise as soon as one transaction ends the next statement automatically starts the next transaction.

As far as DDL or DCL statements are concern they are single statements and are committed automatically with end of transaction

Transaction Control(TCL Commands)

<u>COMMIT:</u> Ends the current transaction by making all pending data changes permanent

ROLLBACK: Ends the current transaction by discarding all pending data changes

Transaction Control (cont.)

<u>Savepoint:</u> Instead of doing complete rollback at any abnormal terminating event we can be partially rollback by setting savepoints in our code.



Example 3: create a new department 'Advertising' with 'Atlanta' as a location with deptno = 50 than assign this department to Mr jones of LoyalEmp table.

INSERT INTO dept (deptno, dname, loc)
VALUES (50, 'ADVERTISING', 'ATLANTA');

UPDATE LOYALEMP

SET DEPTNO = 50
WHERE EMPNO = 7566;

COMMIT;

Step 3

