

TCL – Transaction Control Language

- **Definition:** Transaction એ database ની operation નો set છે. જે કોઈ એક task પરફોર્મ કરે છે.
- Database ની consistency માટે transaction successfully complete થાય અથવા completely fail થવું જોઈએ.
- ઉદાહરણ, **Successfully Complete Transaction:** એક account માંથી બીજી account માં transfer કરવા માટે.
- **Transaction complete** થાય ત્યારે બે accounts માં balance ને update કરવામાં આવે છે: source અને destination.
- તેમાં બે database operation ની જરૂર પડે છે .એક source account માંથી amount debit કરવામાં આવે છે અને destination account માં amount credit કરવામાં આવે છે.
- ઉદાહરણ, **Fail Transaction:** જો fund transfer થતી વખતે source account માંથી amount debit થાય અને destination account માં credit થતી વખતે system failure થાય તો database inconsistent થઈ જાય છે.
- તેથી, balance બંને account માં update થવી જોઈએ અથવા એક પણ account માં થવી જોઈએ નહિ.
- તેથી, આપણે database ના operations ની sequence ને પણ transaction કહેવાય છે.
- આ operation માં insert, update અને delete નો સમાવેશ થાય છે.
- આ operation બે step માં પરફોર્મ થાય છે:
 - 1) ફંક્શન memory માં જ ચેન્જ કરે છે.
 - 2) Changes permanently hard disk માં save થાય છે.
- Transaction ની શરૂઆત commit પછી ના પહેલા SQL statement થી થાય છે અને rollback નો ઉપયોગ કરી ને Transaction ને undone કરી શકાય છે.
- Commit અથવા rollback command ની મદદથી transaction પૂર્ણ કરી શકાય છે જ્યારે transaction પૂર્ણ થાય ત્યારે transaction દરમિયાન જે lock નો ઉપયોગ થયો હોય તે બધા release થાય છે.
- TLC Commands નો ઉપયોગ transaction ને manage કરવા માટે થાય છે જે નીચે આપેલા છે.
 - 1) commit
 - 2) rollback
 - 3) savepoint

Commit :

- Transaction Commit ના બે પ્રકાર છે.
 - 1) Explicitly
 - 2) Implicitly

1) Explicitly Commit:

- Transaction ને Commit કરવા user દ્વારા Commit Command આપવામાં આવે તેને Explicit Commit કહે છે.
- Current Transaction એ Commit Command દ્વારા Terminates થઈ અને બધા permanent changes થઈ hard disk માં save કરે છે
- DML Operation એ insert, update અને delete તેને permanently hard disk માં save કરતું નથી. જ્યાં સુધી Commit Command ન આપવામાં આવે છે.

syntax:

COMMIT;

Output:

Commit Complete

2) Implicit Commit:

- ઘણા એવા operation પણ છે જેમાં Commit automatically oracle દ્વારા કરવામાં આવે છે
- જો Command નીચે આપેલા છે.

1) Quit Command :

- Oracle દ્વારા SQL * PLUS Session disconnect થઈને end થાય છે

2) Exit Command :

- Oracle દ્વારા SQL * PLUS Session Disconnect થઈ ને end થાય છે

3) Data Definition Language (DDL) Command :

- DDL Commands જોવા કે CREATE , ALTER, DROP ની effect transaction દરમિયાન permanently hard disk માં save થાય છે.

ROLLBACK: Canceling a Transaction Completely

- ROLLBACK Command નો ઉપયોગ કરીને transaction completely અથવા partially cancel કરે છે .
- ROLLBACK Command એ current transaction ને terminates કરે છે અને transaction દરમિયાન જે કોઈપણ data base માં change થયા હોય તેને undone કરે છે.
- Oracle માં auto rollback પણ થાય છે. જ્યારે Computer failure થાય અને જ્યારે database ફરીશી open કરવામાં આવે ત્યારે uncommitted work ને oracle authomatically roll back કરે છે .

SAVEPOINT : Cancelling a transaction Partially

- RollBack Command નો ઉપયોગ transaction ને partially પૂર્ણ કરવા માટે થાય છે .

Syntax:

```
ROLLBACK TO SAVEPOINT save_point _name ;
```

Output:

ROLLBACK Complete .

- તે transaction partially રીતે cancel કરવા માટે savepoint create કરવા માં આવે છે.
- Savepoint એ transaction દરમિયાન પોઇન્ટ બનાવે છે.
- નીચે પ્રમાણે આપેલા Savepoint નો ઉપયોગ કરી ને Savepoint બનાવી શકાય છે .

Syntax:

```
SAVEPOINT SAVEPOINT_NAME;
```

Output:

Savepoint Created ;

- જ્યારે Rollback નો ઉપયોગ savepoint સાથે ફરી transaction ને partially cancel કરી શકાય છે.
- Savepoint Create કર્યી પછી ના બધા operation undone થાય છે.
- કોઈપણ transaction માં એક કરતા વધારે savepoint બનાવી શકાય છે.

Data Control Language-DCL

- Database ની security એ DBMS માં મહત્વ નું છે.
- Unauthorized person database ને access કરી શકતા નથી.
- database content ને ssecca કરવા user ને rights આપવામાં આવે તેને **privileges** કહે છે
- Oracle બે phase માં database content ને security આપે છે:
 - users database ને access કરવા માટે valid user, id, password ની જરૂર પડે છે અને પછી login કરી શકે છે .
 - user દ્વારા database ને access કરવા માટે ના �priviledges આપેલા હોવા જોઈએ.
- Multi-user system માં જુદા-જુદા users database ના જુદા-જુદા parts ને access કરવાની જરૂર પડે છે.
- database designer એ નક્કી કરે છે કે user database માંથી કયો part access કરશે.
- જુદા-જુદા users ને જુદા-જુદા privileges (permission) આપવામાં આવે છે.

- ઉદાહરણ: banking system માં, Customers એ ફક્ત પોતાની જ એટલે તેના જ account ની માહિતી ને જોઈ શકે છે પરંતુ તેમાં કોઈપણ change કરી શકતા નથી.
- કોઈપણ **customer, account** ની **balance change** કરી શકતા નથી અને બીજા customer ના account ને access પણ કરી શકતા નથી.
- User database objects જેવા કે, tables બનાવે છે. જો કોઈપણ user ને કોઈ બીજા user દ્વારા બનાવામાં આવેલા objects access કરવા હોય તો તે object ના owner દ્વારા તે user ને permission આપવામાં આવે છે.
- જુદા-જુદા Database object ના access ને control કરવા Oracle માં બે commands છે: **GRANT** અને **REVOKE**.

GRANT: Granting Privileges

- Grant Command નો ઉપયોગ કોઈપણ users ને database object અથવા database object ના part ને access કરવા માટે ની permission આપે છે .
- Grant Command database object જેવા કે, table, sequence અને view ને જુદી-જુદી રીતે access કરવાની permission આપે છે.

Syntax:

```
GRANT Object _ Privileges
    ON Obeject _ name
    TO usres _ name
    [WITH GRANT OPTION];
```

- database object નો owner એ બધા જ privileges અથવા specific privileges કોઈ બીજા user ને આપી શકે છે .
- with Grant Option નો ઉપયોગ user ને grantee બનાવા માટે થાય છે તે user બીજા user ને privileges આપી શકે છે.
- User બધા જ અથવા specific privileges બીજા user ને આપી શકે છે.
- જુદા-જુદા object privileges નીચે મુજબ છે:

prilege	allows user...
all	નીચે આપેલા બધા operation પરફોર્મ કરી શકે છે.
alter	Alter command દ્વારા table structure બદલી શકે છે.
delete	Delete command દ્વારા tabel ના recode delete કરી શકે છે.

index	Create index command દ્વારા table માં index create કરી શકે છે.
insert	Insert command દ્વારા table માં recode insert કરી શકે છે.
references	foreign key બનાવીને reference table ને reference આપી શકે છે.
select	select Command નો ઉપયોગ કરી શકે છે.
update	Update command દ્વારા table ની record ને modify કરી શકે છે.

- **Example:** ધ્યારો કે ત્રણ user - user1, user2 અને user3 છે. Table – **Customer** નો owner user1 છે. user1 તરીકે user2 ને Customer table પર બધા જ �data manipulation operation ની permission આપવામાં આવે અને user2 ને permission ને બીજા users ને પણ આપી શકે

Input:

```
GRANT ALL
ON customer
TO user2
WITH GRANT OPTION;
```

Output:

Grant succeeded.

- જો grant option ન આપવામાં આવે તો user2 ને બધા privileges હોય પરંતુ એ privileges ને બીજા users ને આપી શકે નહિએ.
- **Example:** user2 તરીકે, user3 ને select અને insert privileges આપો

Input:

```
GRANT SELECT, INSERT
ON user1.customer
TO user3;
```

Output:

Grant succeeded

- **Example:** user2 તરીકે, Customer table ની બધી માહિતી display કરો.

Input: SELECT * from user1.customer.

- Customer table પર કોઈપણ operation perform કરવા માટે, user2 ને suffix તરીકે table ની owner name ની જરૂર પડે છે

REVOKE –Revoking Privileges

- Revoke Privileges means જેટલા Privileges આપેલ હોય તેને પાછા લઈ શકાય છે.
- Object નો જે owner છે એ જેટલા privileges આપેલ હોય તેને પાછા લઈ શકે છે. user કે જે database object નો owner નથી તો પણ તેને આપેલા privileges ને પાછા લઈ શકાય.

Syntax:

```

Revoke object privileges
ON      object name
FROM    user name;
```

Example: User2 તરીકે, user3 પાસે થી select અને insert ની privileges revoke કરવા માટે નીચે મુજબ ની syntax નો ઉપયોગ થાય છે.

Input:

```

Revoke SELECT,INSERT
On      user1.customer
From    user3;
```

Output:

Revoke succeeded

LOCK

- Transaction એ database operation નો set છે. જે particular task પરફોર્મ કરે છે.
- Transaction માં single SQL statement નો સમાવેશ થાય તો તેને single query transaction (SQT) કહે છે.
- Transaction માં multiple sql statement નો સમાવેશ થાય તો તેને multiple query transaction (MQT) કહે છે.
- કોઈપણ database system માટે તેની જરૂરિયાત પ્રમાણે એક કરતા વધારે user એકસાથે database ને access કરી શકવા જોઈએ.
- જ્યારે multiple user data concurrently access કરતા હોય ત્યારે data integrity ની જાળવવી ખૂબ જ જરૂરી છે.
- જ્યારે multiple users એક સાથે database access કરતા હોય ત્યારે data ને protect કરવા માટેની technique ને Concurrency Control કહે છે.
- Oracle માં **locking** method નો ઉપયોગ કરીને Concurrency Control implement થઈ શકે છે
- Lock એટલે બીજા user માટે data access કરવા માટે restrict કરવામાં આવે છે.
- Lock mechanism નો ઉપયોગ કરીને data integrity જાળવવામાં આવે છે.
- Locking ના બે પ્રકાર છે:
 - Implicit locking
 - Explicit locking

Implicit locking

- જ્યારે sql statement નું executing થતું હોય ત્યારે table માં રહેલા data automatically locked થઈ જાય છે તેમાં user interference ની જરૂર નથી આવા પ્રકાર ના locks નો implicit locks કહેવાય છે.
 - જ્યારે આ lock data પર apply થાય ત્યારે oracle માં બે issue થાય છે:
 - 1) type of lock
 - 2) level of lock
 - જુદા-જુદા પ્રકાર ના અને લેવલ ના lock નીચે પ્રમાણે છે:
 - 1) types of lock:
 - o oracle બે પ્રકાર ના lock નો ઉપયોગ થાય છે: shared lock અને exclusive lock
- ❖ Shared lock:
- o જ્યારે read operation પરફોર્મ થતું હોય ત્યારે shared lock apply થાય છે.
 - o Read operation data display કરવા માટે થાય છે તેમાં Select statement નો ઉપયોગ થાય છે.
 - o Multiple shared locks એ table અથવા બીજા object મા એકસાથે place થઈ શકે છે.
 - o Read operation એ table data ને modify કરતું નથી તેથી multiple read operation એ data integrity મા કોઈપણ problem કરતું નથી.
- ❖ Exclusive lock:
- o જ્યારે write operations perform થાય છે ત્યારે exclusive lock apply થાય છે.
 - o INSERT, UPDATE અથવા DELETE statement નો ઉપયોગ કરીને write operations થી data modify કરી શકાય છે.
 - o માત્ર એક જે exclusive lock table અથવા બીજા object મા place થાય છે.
 - o Write operation table data ને modifies કરે છે. એક કરતા વધારે write operation data integrity પર અસર કરે છે અને database ને inconsistent બનાવે છે.
 - o તેથી multiple users એક સાથે એક જે data ને modify કરી શકતા નથી.
- ❖ Deadlocks :
- o Transaction નો set deadlock કહેવાય જો set ના દરેક transaction lock માટે wait કરે છે જે કોઈ set ના બીજા transaction દ્વારા occupy થયેલ હોય.
 - o અહીંયા દરેક transaction એ lock માટે wait કરે છે કોઈ પણ transaction એક પણ lock release કરતા નથી. તેથી બધા transaction lock માટે wait કરે છે.
 - o તેથી, બધા જે transaction wait કર્યો કરે છે.

- Process मां बे अथवा वधारे threads deadlock थाय जो नीचे नी कोईपण condition साची पડ़े:
 - Transaction मां पहेले थी ४ locks ने hold करेला होय अने नवा locks नी request करे.
 - New lock माटे नी request एक साचे करवामां आवे छे.
 - बे अथवा वधारे transaction ज्यारे Circular chain बनावे के जेमां ४ transaction ए lock माटे wait करे छे जे chain ना बीजा transaction द्वारा hold थयेला होय छे.
- **Example:** Transaction – A ए data X Lock करेल छे अने Y data पर lock माटे wait करे छे. ज्यारे Transaction – B ए data Y Lock करेल छे अने X data पर lock माटे wait करे छे. आ परिस्थिति मां एक पण transaction आगाज वधी शके नही.

2) Level of Lock:

- एक करता वधारे users ने एक ४ table ना जुदा-जुदा भाग access करवानी ज़रूर पडे छे.
- उदाहरण तरीके, 'xyz' ब्रांच नो मेनेजर फक्त 'xyz' ना ४ Accounts ने access करवानी ज़रूर होय छे. ज्यारे बीजा मेनेजर बीजा Accounts access करी शके.
- तेथी, जो एक ४ user द्वारा table lock थयेल होय तो बीजा user ने table ना बीजा record access करवा होय तो पण wait करवो पडे छे.
- आ प्रकार ना प्रोब्लेम ने दुर करवा अने वधारे concurrent access शक्य बने ते माटे ज़रूरीयात प्रमाणे table पर अथवा table ना अमुक भाग मां ४ lock place करवामां आवे छे.
- तेथी, कोई पण customer पोतानुं account ४ access करी शके छे अने बीजा customer तेनुं account access करी शके.
- Oracle ए त्रण level ना implicit lock आपे छे.
- आ levels ए SQL statement मां रहेला WHERE clause पर आधार राखे छे

❖ ROW LEVEL LOCK:

- ज्यारे WHERE clause एक ४ row आपे छे त्यारे आ lock नो उपयोग थाय छे.
- Example : WHERE student_ID =1;
- आशी conditionमां मात्र single row ४ lock थाय छे. Table ना बीजा record ए कोई बीजा user द्वारा access थई शके छे.

❖ PAGE LEVEL :

- WHERE clause एक करतां वधारे rows आपे छे त्यारे आ lock नो उपयोग थाय छे.

EXAMPLE : WHERE B_name ='CE';

- આ condition માં એક કરતા વધારે rows lock થાય છે અને table ના બીજા records એ કોઈ બીજી users દ્વારા access થઈ શકે છે .
- ❖ TABLE LEVEL :
 - જ્યારે SQL statement માં WHERE condition ન હોય ત્યારે આ lock નો ઉપયોગ થાય છે.
 - આ condition માં query એ entire table ને access કરે છે તે entire table lock થાય છે જેથી, બીજો કોઈ પણ user table ના બીજા part ને access કરી શકે નહીં .
- Implicit lock માં એક જ ટેબલ પર કોઈ user દ્વારા exclusive lock apply થયેલ હોય તો પણ બીજા કોઈપણ user shread lock apply કરી શકે છે.
- જેથી, જો કોઈ user content change કરે તો તે બીજા user જોઈ શકે છે પરંતુ તે older data read કરશે.

Example:

- જો કોઈ user Account table ના કોઈ account ની balance update કરે પરંતુ, changes ને permanently **COMMIT** command દ્વારા save ન કરેલ હોય. હવે, કોઈ user table ના contents ને read કરશે તો તેને updated balance ને બદલે older data મળશે.
- તેનું કારણ એ છે કે change ને હજુ permanently hard disk માં save કરેલ નથી.
- આ પ્રકાર ના પ્રોબ્લેમ ને દુર કરવા **explicit lock** નો ઉપયોગ થાય છે.

EXPLICIT LOCKS :

- Users દ્વારા table પર કે table ના કોઈ પણ ભાગમાં lock apply કરે તો તેને Explicit lock **Explicit locks** કહે છે. તેમાં user interference ની જરૂર પડે છે.
- Table ના owner દ્વારા table પર explicit lock place કરી શકાય છે. બીજા users પણ explicit lock place કરી શકે છે જો તેને privilege હોય.
- Oracle માં explicit lock એ implicit lock ને override કરે છે.
- Entire table અથવા table ના records ને explicitly lock કરી શકાય છે જેના માટે બે commands નો ઉપયોગ થાય છે:

1) The SELECT..... FOR UPDATE statement

Syntax : **SELECT * FROM tableName FOR UPDATE [NOWAIT];**

- આ statement નો ઉપયોગ કરી ને records ને updates કરવા માટે exclusive locks place કરી શકાય છે.
- SELECT statement માં WHERE clause ને આધારે lock નું level apply થાય છે.

- જો NOWAIT આપેલ ન હોય અને જો table પહેલે થી જ બીજા user દ્વારા lock થયેલ હોય તો આ command wait કરશે જ્યાં સુધી lock release ન થાય.
- પરંતુ, જો NOWAIT આપેલ હોય અને table free ન હોય તો, આ command એ error message "Resource is busy" return કરે છે.
- જ્યારે commit અથવા rollback execute થાય ત્યારે lock release થાય છે
Example: user1 તરીકે, 'XYZ' બ્રાંચ ના accounts પર SELECT...FOR UPDATE statement નો ઉપયોગ કરીને exclusive lock place કરો.

```
SELECT * FROM Account WHERE B_name = 'XYZ' FOR UPDATE;
```

Example: user2 તરીકે, 'XYZ' બ્રાંચ ના બધા accounts માં 1000 balance update કરો.

```
UPDATE Account SET balance = 1000 WHERE B_name = 'XYZ';
```

- આ case માં user2 wait કરશે જ્યાં સુધી user1 commit અથવા rollback નો ઉપયોગ કરી lock release ન કરે.

2) The lock table statement :

Syntax : LOCK TABLE tableName IN lockMode MODE [NOWAIT];

- આ statement માં table માં આપેલા ચોક્કસ mode ને આધારે lock place થાય છે .
- જો NOWAIT દર્શાવેલ હોય અને એ table free ન હોય તો આ command error message "Resource is busy" return કરે છે.
- જુદી - જુદી lock ના mode નીચે પ્રમાણે દર્શાવેલ છે .

MODES	SPECIFIES
EXCLUSIVE	DML query ને table પર allow કરે પરંતુ બીજા operation allow કરશે નહિં. બીજા user table ના data માત્ર display કરી શકે છે.
SHARED	એક સાથે એક કરતાં વધારે user દ્વારા select ની query ને allow કરે છે DML operation ને allow કરશે નહીં.
ROW SHARED	Row level lock દર્શાવે છે. User એ table ને lock કરી શકતા નથી. Concurrent access બધા user માટે allow કરે છે.
SHARE UPDATE	ઉપર મુજબ જ છે પરંતુ જુના version સાથે compatible છે .
ROW EXCLUSIVE	ROW SHARE જેવું જ છે, પરંતુ shared lock allow કરશે નહિં. તેથી, એક સમયે એક table માં એક user access કરી શકે છે.

Example: User1 તરીકે, Account table પર LOCK TABLE Statement નો ઉપયોગ કરી exclusive lock place કરો.

```
LOCK TABLE Account
IN EXCLUSIVE MODE [ NOWAIT];
```

Example: User1 તરીકે, Account 'A01' ની balance 1000 થી update કરો.

```
UPDATE Account SET balance = 1000 WHERE A_No = 'A01';
```

- હવે, user2 તરીકે table ના બધા content ને display કરો અને 'A01' ની balance જુઓ. તે જુની balance દેખાડશે. કારણ કે, user1 દ્વારા update operation ને commit કર્યું નથી.
- User1 તરીકે commit કર્યું પછી, user2 તરીકે account table ને display કરવામાં આવે તો updated balance જોવા મળે છે.
- અહિયા, user2 એ account table ના data ને જોઈ શકે છે. કારણ કે, EXCLUSIVE mode માં lock કરેલ છે. જ્યારે બીજા user data ને જોઈ શકે છે. પરંતુ, બીજા operation allow કરશે નહિએ.
- તેથી user2 એ table ના content ને modify કરી શકતા નથી. જ્યાં સુધી lock release ન થાય.

Difference between DBMS and RDBMS :

DBMS	RDBMS
✓ Tables વચ્ચે relationships ન હોય.	✓ Database માં અન્ય table સાથે relation ધરાવતા table માં data store કરે છે તેને RDBMS કહે છે.
✓ Data flat file માં store હોય એટલે બીજા data સાથે સબંધ ન હોય.	✓ Data table ના ફોરમમાં store હોય છે અને tables વચ્ચે સબંધ હોય છે
✓ Normalization process ની જરૂર પડતી નથી.	✓ Database table consistency જાળવવા માટે normalization process ની જરૂર પડે છે.
✓ DBMS client-server architecture ને support કરતું નથી.	✓ RDBMS client-server architecture ને support કરે છે.
✓ તે data manipulation ના સદર્ભમાં કોઈપણ integrity constraints ની જરૂર પડતી નથી.	✓ અહીં integrity constraints ની જરૂર પડે છે.
✓ તે simpler business application માટે ઉપયોગી બને છે.	✓ તે વધુ complex application માટે ઉપયોગ થાય છે.

VIEWS :

- View એક વરાચ્યુલ અથવા લોજિકલ table છે કે જે table ના ભાગ ને display કરવા અથવા manipulate કરવાની permission આપે છે.
- જે table પરથી view બનાવવામાં આવે તેને base table કહે છે .
- View એ normal table ની જેમ જ કામ કરે છે. બંને વચ્ચે તફાવત એ છે કે table ની જેમ view ને store કરવા storage space ની જરૂર પડતી નથી.
- View એ select query દ્વારા બનાવવામાં આવે છે. જે base table નો ઉપયોગ કરે છે.
- કોઈપણ operation view પર કરવામાં આવે તેની અસર base table પર થાય છે .
- View અને normal table વચ્ચેનો તફાવત storage space નો છે. database માં માત્ર view ની વ્યાખ્યા store કરવા જોટલી જ જોયા રોકે છે.
- કોઈપણ operation view પર કરવામાં આવે તો તે ખરેખર base table પર થાય છે.
- ઉદાહરણ તરીકે, select operation view પર કરવામાં આવે તો base table નો content display કરે છે. તેવી જ રીતે, Insert, Update, Delete operation base tableના content ને modify કરે છે .

VIEW ના પ્રકારો:

- View ને સામાન્ય રીતે 2 ભાગમાં વહેંચી શકાય છે:
 - 1) **READ-ONLY VIEW :**
 - માત્ર select operation મંજૂરી આપે છે. આનો અર્થ છે કે ફક્ત data જોઈ શકાય છે. તેને modify કરી શકાય નહીં.
 - INSERT, UPDATE કે DELETE Operation allow નથી આનો મતલબ કે base table ના content ને modify ન કરી શકાય.
 - 2) **UPDATABLE VIEW :**
 - SELECT operation સાથે સાથે INSERT, UPDATE અને DELETE Operation allow કરે છે. આનો મતલબ કે base table ના content ને display સાથે – સાથે modify પણ કરી શકાય છે.

Create VIEW:

- View ને બનાવવા માટે નીચેની syntax નો ઉપયોગ થાય છે.

Syntax :

```
CREATE [OR REPLACE] VIEW viewname
```

```
As SELECT ....
```

```
[WITH READ ONLY];
```

- આ ll statement માં SELECT statement માં આપેલ Query ના આધારે એક view બનાવે છે .
- OR REPLACE નો ઉપયોગ જો view પહેલે થી બનેલ હોય તો તેને Overwrite કરી નવો view બનાવે છે.
- WITH READ ONLY Option read only view બનાવે છે.
- જો આ option લખવામાં ન આવે તો by default updatable view બને છે.
- SELECT Statement સાથે WHERE ,ORDER BY ,GROUP BY નો ઉપયોગ કરી શકાય છે.
- એક અથવા એક કરતા વધારે base tables નો ઉપયોગ કરીને view બનાવી શકાય છે.
- અહિયા, બે table Account ને Branch નો ઉપયોગકરીને view કેવી રીતે બનાવી શકાય તે દર્શાવેલ છે.

Account			Branch	
Acc_No	Balance	B_Name	B_Name	B_Address
A01	1000	Rjt	Rjt	Kalavad Road, Rajkot
A02	2000	Ahmd	Ahmd	Elisbridge, Ahmedabad
A03	3000	Srt	Srt	Mota Bazar, Surat

❖ એક Base table નો ઉપયોગ કરી ને view નીચે મુજબ બનાવી શકાય:

Example 1: Account table નો ઉપયોગ કરી ને 'Acc_Rjt' View બનાવો.

Input: CREATE VIEW Acc_Rjt

```
AS SELECT * FROM account WHERE B_Name ='Rjt';
```

Output:

View created

- કોઈપણ SQL statement table ની જેમ જ view નો ઉપયોગ કરી શકાય છે.
- તેના માટે table ના નામ ને બદલે view ના નામ નો ઉપયોગ કરવામાં આવે છે.

Example 2: 'Acc_Rjt' view ના બધા record display કરો.

Input:

```
SELECT * FROM Acc_Rjt;
```

Output:

Acc_No	Balance	B_Name
A01	1000	Rjt

Example 3: Account and Branch table નો ઉપયોગ કરી ને 'Acc_branch' view બનાવો. જેમાં બધા account અને તેની branch ની માહિતી હોય.

Input :

```
CREATE VIEW Acc_branch
AS SELECT Acc_no, balance, account.b_name branch, b_address
FROM account, branch WHERE account.b_name = branch.b_name;
```

Output:

View created

Example 4: 'Acc_Branch' view ની માહિતી display કરો.

Input:

```
Select * from acc_branch;
```

Output:

Acc_no	balance	branch	B_address
A01	1000	Rjt	Kalawad road, Rajkot
A02	4000	Ahmd	Elisbridge, Ahmedabad
A03	3000	Srt	mota bazar, surat

Alter View

- Alter view નો ઉપયોગ View માન્ય છે કે અમાન્ય એ check કરવા માટે થાય છે

Example: View acc _ branch ને ફરીથી compile કરો.

Syntax:

```
ALTER VIEW Acc_Branch Compile;
```

- જો oracle database Acc_Branch ને ફરીથી compile કરતી વખતે કોઈ error ન આવે તો Acc_Branch માન્ય ગણાય છે અને જો error આવે તો acc _ Branch અમાન્ય ગણાય છે.

view ના ફાયદાઓ

- object મા data store કર્યા વિના data display કરી શકાય છે.
- બે અથવા વધુ table ને join કરી અને user ને એક object તરીકે દર્શાવી શકાય છે.
- Table ની column ને hide કરી શકાય છે.
- View માં security આપી શકાય.
- View ના બે મુખ્ય advantages છે.

1) Flexible enforcement of Security

- view નો ઉપયોગ કરી ને table data માંથી ફક્ત અમુક મર્યાદિત data જુદા-જુદા user ને આપી શકાય છે.

- **Example:**

- બધી બ્રાંચ ના મેનેજર તેની જ બ્રાંચ ના Account અને customer ની માહિતી access કરી શકવા જોઈએ.
- તેના માટે બધી બ્રાંચ માટે જુદા-જુદા table બનાવાની જરૂર પડે છે. જેને લીધે data redundancy અને database ની design complex થાય છે.
- આ પ્રોબ્લેમ ને દુર્ગત કરવા માટે જુદા-જુદા table ને બદલે જુદા-જુદા view base table પર બનાવી ને table પર rights આપવાને બદલે view ને rights આપી શકાય છે.
- Example-1 માં 'Acc_Rjt' view માં ફક્ત 'Rjt' બ્રાંચ ના જ account ની માહિતી છે.

Example-5: 'Rjt_manager' ને 'Acc_Rjt' view પર બધા privileges આપો અને તે આ privileges બીજા user ને પણ આપી શકે.

Input: GRANT ALL

```
ON Acc_Rjt
TO Rjt_Manager
WITH GRANT OPTION;
```

Output:

Grant Success.

2) Simplification of Complex Query:

- Complex Query ને સરળ બનાવી શકાય છે.

- **Example:**

- account 'A01' ની branch અને branch address display કરવા માટે acc_branch ના view પર નીચે મુજબ query fire કરી શકાય છે .

Input:

```
SELECT branch, b_address FROM acc_branch WHERE acc_no='A01';
```

Output:

Branch	B_Address
Rjt	Kalawad Road, Rajkot

- અહીં 'acc_branch' account અને branch tables ને જોડી ને બનાવા માં આવે છે પરંતુ તે એક અલગ table ની જેમ જ કાર્ય કરે છે અને table ને combine કરે છે.

View ના ગેરફાયદાઓ

- જ્યારે base table ના structure માં change થાય ત્યારે view inactive બની જાય છે.
- view એક object છે તેથી તે space રોકે છે.

Destroying a view

- Drop view command view ને drop કરે છે.

- view destroying થાય છે તો base table પર અસર થશે નહિં.
- જો કોઈ base table ને drop કરવામાં આવે અથવા view માં રહેલ �column બદલાયેલ હોઈ તો view valid રેહશે નહિં.

Syntax:

```
DROP View view_name;
```

Example 7: Acc_Branch ને drop કરો.

Input: Drop View acc_branch;

Output: view dropped

INDEXES

- જ્યારે data ને search કરવા માટે data કોઈ specific order માં sort કરેલા હોય છે (accesending અથવા desending) ત્યારે search કરવું સરળ બને છે.
- જ્યારે record sorted ન હોય ત્યારે serach કરવા માટે કોઈ પણ query fire કરવામાં આવે ત્યારે તે લાઇન માં બધા record ને એક પછી એક ચેક કરે છે
- નીચેનું table આ પરિસ્થિતિ ને દર્શાવે છે અહિયાં customer table એ sorted નથી તેમાં customer 'sophia' નો record search કરવાનો છે તેથી આપણે બધા record એક પછી એક લાઇન માં ચેક કરવા પડે છે.

Customer

Name	CID	Address	City
Jack	C01	Corporation street	Manchseter
Emily	C02	Council Area	Perth
Sophia	C03	Corporation street	Manchseter
Scott	C04	Sardar colony	Anand

- તેથી, search ને efficient બનાવા માટે data sorted હોવા જોઈએ.
- પરંતુ, તેના માટે દરેક insert, update અથવા delete operation માટે record ને sort કરવાની જરૂર પડે છે. અને Sorting ની પ્રક્રિયા time consume કરે છે.
- Search જુદી-જુદી column દ્વારા કરવામાં આવે છે. તેથી, table ને દરેક column પર sort કરી શકાય નહિં.
- Oracle એક object આપે છે, જેને index તરીકે ઓળખાય છે જે ઉપર ના problem ને સરળતાથી દુર કરે છે અને search ને સરળ બનાવે છે.
- Index એ table માં રહેલ column ના content નું ordered list છે.
- Index એ table જેવું છે તેને ઓછા માં ઓછી બે column હોય છે.
 - column કે જેમાં sorted data હોય છે જેના પર index create થાય છે

- 2) column કે જેમાં table ની દરેક row માટે row id દર્શાવે છે
- ROWID એ દરેક નવી row insert થાય તેના માટે નો unique identifier છે.
 - નીચેના table માંથી 'shopia' customer search કરવામાં આવે છે.
 - અહિયા, ROWID ને સરળતા માટે 3 digit થી દર્શાવેલ છે:

Customer					Index	
ROWID	Name	CID	Address	City	Name	ROWID
001	Jack	C01	Kalawad Road	Rajkot	Jack	001
002	Emily	C02	C.G.Road	Ahmedabad	Emily	002
003	Shopia	C03	Kamati Baug	Vadodara	Shopia	003

- જ્યારે આ table પર query fire કરવામાં આવે છે ત્યારે table ને બદલે જે Column પર index બનાવેલ છે તેના આધારે search process થાય છે.
- Index કમસર search કરે છે અને record મેળવવા માટે ROWID જરૂરી છે.

ફાયદાઓ

- Index જ્યારે કોઈ બે કોલમ પર બને ત્યારે index માં insert , update. અને delete operation માટે વધારે time લાગતો નથી.
- Index બનાવેલ column માં data sorted હોય છે. જેશી search ની process fast થાય છે.

ગેરફાયદાઓ

- DML operation ને slow કરે છે.
- Indexes queries ની execution ને slow કરે છે.

RowID

- RowID એ table ના દરેક record નું યુનિક identifier છે.
- RowID hexadecimal string છે, જે database માં રહેલા record નું લોજિકલ એડ્રેસ દર્શાવે છે.
- Oracle માં pseudo column ROWID છે, જે table મા રહેલા record નું RowID આપે છે
- RowID નો ઉપયોગ બીજુ column ની જેમ જ select statement માં થાય છે.
- RowID ના બે format છે:

1) Extended:

- તે 18 digit ની string છે અને 00000FFFFBBBRRRRRR form માં હોય છે.
- આ ફોર્મેટ oracle 8i અથવા higher versions મા ઉપયોગ થાય છે.

2) Restricted:

- તે 15 digit ની string છે અને BBBB.BBBBB.FFF form માં હોય છે.
- આ ફોર્મેટ oracle 7 અથવા તેની પહેલાના version મા ઉપયોગ થાય છે.

- String માં આપેલ character નો અર્થ નીચે table માં દર્શાવેલ છે:

Character	Specifies
000..	Data object નંબર database segment દર્શાવે છે.
Fff...	Data file નંબર data જે file માં આવેલ છે તે દર્શાવે છે.
BBB..	Data file આવેલ block નંબર એ જે block માં record આવે છે તે દર્શાવે છે.
RRR...	Block માં રહેલ record નો નંબર દર્શાવે છે.

- Table માં insert થયેલ records નું ગ્રુપ બની ને Block બનાવે છે. આ Block નું ગ્રુપ બની ને data file બને છે.

ફાયદાઓ

- એકવાર index પરથી ROWID મળ્યા બાદ, તે data file, block અને record નંબર દર્શાવે છે. જેથી, record ને સીધો access કરી શકાય છે.
- તેથી, hard disk માંથી data ને search કરવા માટે ઓછો time લાગે છે.
- જેથી data retrieval time માં ઝડપ થી સુધારો થાય છે.

Example 8: Customer table બધી column સાથે ROWID પણ ડિસ્પ્લે કરો:

Input: Select RowID, Name, CID, Address, City from Customer;

Output:

RowID	Name	CID	Address	City
Aaafx7aabaaakjcaa	Jack	C01	Corporation street	Manchseter
Aaafx7aabaaakjcaab	Emily	C02	Council Area	Perth
Aaafx7aabaaakjcaac	Sophia	C03	Corporation street	Manchseter
Aaafx7aabaaakjcaad	Scott	C04	Sardar colony	Anand

Type of index

- Index નાલ 4 type છે:
 - 1) Duplicate indexes
 - 2) Unique indexes
 - 3) Simple indexes
 - 4) Composite indexes
- Duplicate Index અને Unique index વચ્ચે નો તફાવત નીચે મુજબ છે:

Duplicate Index	Unique Index
<ul style="list-style-type: none"> Index column માં જો duplicate વેલ્યુ allow કરે તો તેને Duplicate Index કહે છે. 	<ul style="list-style-type: none"> Index column માં જો duplicate વેલ્યુ allow ન કરે તો તેને Unique Index કહે છે.
<ul style="list-style-type: none"> Duplicate index, primary key અથવા unique key column પર બનાવી શકાય 	<ul style="list-style-type: none"> Unique index, primary key અથવા unique key column પર બનાવી શકાય.

નહિ. કારણ કે, તેમાં duplicate વેલ્યુ આવી શકે નહિ.

- **Simple Index:** જો index એક જ �column પર બનાવામાં આવે તો તેને Simple Index કહે છે.
- **Composite Index:** જો index એક કરતા વધારે column પર બનાવામાં આવે તો તેને Composite Index કહે છે.

Creating an Index :

- Simple Index બનવા માટે ની Syntax નીચે મુજબ છે:

Syntax:

```
CREATE [UNIQUE] INDEX index_name
ON table_name (column name);
```

- Bydefault index duplicate index બને છે.
- જો unique option આપવામાં આવે તો unique index બને છે.

Example 9: Customer table માં 'Name' column પર Simple index બનાવો:

Input:

```
CREATE INDEX ind_cust_name
ON Customer (Name);
```

Output:

index created.

- Composite Index બનવા માટે ની Syntax નીચે મુજબ છે:

Syntax:

```
CREATE [UNIQUE] INDEX index_name
ON table_name (columnName 1, columnName 2,...);
```

- જો index બનાવતી વખતે એક કરતા વધારે column આપવામાં આવે તો composite index બને છે નહીંતર Simple Index બને છે .
- જો એ કરતા વધારે column પર index બનાવામાં આવે તો પહેલા ની index column માં duplicate data હોય તો જ બીજી index column ને consider કરવામાં આવે છે .
- ઉપર ની syntax મુજબ જો first column માં duplicate data રાખવામાં આવે તો જ sorting second column ને આધારે થાય છે .

Multiple Index On Table

- એક table માં એક કરતા વધારે index જુડી-જુડી column પર હોય શકે છે.

- જો table માં એક કરતા વધારે index હોય તો where અથવા order by clause ને આધાર �index select કરવામાં આવે છે. નહીંતર એક પણ index નો ઉપયોગ થશે નહિં.
- ગેરફાયદા:
 - જો એક table માં વધારે index બનાવામાં આવે તો પરફોર્મન્સ માં સુધારો થવાને બદલે બગડે છે.
 - Index column નાલ content ને sort કરવા માટે time લાગે છે.
 - જો વધારે index બનાવામાં આવે તો content sort કરવામાં time લાગે છે જેને લીધે system slow થાય છે.
 - તેથી, index એવી column પર બનાવામાં આવે છે જેનો ઉપયોગ વારંવાર data મેળવવા માટે થતો હોય.

Destroy an Index

Syntax:

```
DROP INDEX index_name;
```

- આ command નો ઉપયોગ કરીને index ને drop કરવામાં આવે છે.

Example 10: ind_cust_name નામની index ને drop કરો.

Input: `DROP INDEX ind_cust_name;`

Output: Index Dropped

Sequence

- Table ના record ને એકબીજા થી જુદા પાડવા માટે બધા record માં unique વેલ્યુ હોવી જોઈએ.
- Primary key constraint નો ઉપયોગ કરીને column માં duplicate અથવા NULL વેલ્યુ insert થતી અટકાતી શકાય છે.
- આ પ્રકાર ની column માં કમસર વેલ્યુ જેવી કે, 1, 2, 3, અથવા કમસર વેલ્યુ ને string સાથે combine કરી શકાય. જેવી કે, 'A01', 'A02',
- જ્યારે data ને manually insert અથવા update કરવામાં આવે ત્યારે આ પ્રકાર ની sequence ને track કરવી ધારી અધરી છે.
- Oracle નો object Sequence નો ઉપયોગ કરીને આ પ્રક્રિયા ને સરળ બનાવી શકાય છે.
- Sequence એ એક automatic counter છે જે જ્યારે જરૂર હોય ત્યારે sequence નંબર જનરેટ કરે છે.
- Sequence નંબર જનરેટ થાય તેમાં વધારે માં વધારે 38 digit હોય છે.
- Sequence નીચે મુજબ ના કાર્યો કરવા માટે બનાવામાં આવે છે.
 - ✓ ચારતા અથવા ઉત્તરતા કમમાં નંબરો generate કરવા માટે.
 - ✓ નંબરો વચ્ચે interval આપવા માટે.
 - ✓ Advance માં sequence નંબર generate કરવા માટે.

Creating a Sequence

Syntax:

```
CREATE SEQUENCE sequenceName
[ START WITH n
  INCREMENT by n
  MINVALUE n / NOMINVALUE
  MAXVALUE n / NOMAXVALUE
  CYCLE / NOCYCLE
  CACHE n / NOCACHE
  ORDER / NOORDER ];
```

- જો કોઈપણ option વગાર sequence બનાવામાં આવે તો default sequence 1 થી શરૂ થાય અને કમશા ચડતા કમમાં 1 નો વધારો કરે છે.
- Sequence માં નીચે આપેલ table માં રહેલ option નો ઉપયોગ કરી શકાય છે:

Option	Specifies...
START WITH	<ul style="list-style-type: none"> પ્રથમ sequence નંબર ને દર્શાવે છે. ચડતા કમમાં sequence માટે ઓછા માં ઓછી વેલ્યુ : 1 ઉત્તરતા કમમાં sequence માટે વધારે માં વધારે વેલ્યુ : -1
INCREMENT BY	<ul style="list-style-type: none"> Sequence number ની વચ્ચે નો interval દર્શાવે છે. તે કોઈ પણ positive અથવા negative વેલ્યુ હોય શકે છે. પરંતુ 0 વેલ્યુ ન હોય શકે. Default વેલ્યુ 1 હોય છે.
MINVALUE	<ul style="list-style-type: none"> Sequence ની ઓછા માં ઓછી વેલ્યુ દર્શાવે છે.
NOMINVALUE	<ul style="list-style-type: none"> ચડતા કમની sequence માટે ઓછા માં ઓછી વેલ્યુ 1 અને ઉત્તરતા કમમાં sequence માટે ઓછા માં ઓછી વેલ્યુ 10^{-26} હોય છે.
MAXVALUE	<ul style="list-style-type: none"> Sequence ની વધારે માં વધારે વેલ્યુ દર્શાવે છે.
NOMAXVALUE	<ul style="list-style-type: none"> ચડતા કમમાં sequence માટે વધારે માં વધારે વેલ્યુ 10^{27} અને ઉત્તરતા કમમાં sequence માટે વધારે માં વધારે વેલ્યુ 1 હોય છે.
CYCLE	<ul style="list-style-type: none"> Sequence Maximum વેલ્યુ સુધી પહોંચી ને પછી ફરીથી શરૂઆત થી Sequence નંબર જનરેટ કરે છે.
NOCYCLE	<ul style="list-style-type: none"> Sequence maximum વેલ્યુ સુધી પહોંચી ને પછી એક પણ sequence નંબર જનરેટ થતો નથી.
CACHE	<ul style="list-style-type: none"> Advance માં મેમરી માં કેટલી વેલ્યુ જનરેટ કરવી તે દર્શાવે છે. તેના

	માટે ઓછા માં ઓછી વેલ્યુ 2 છે.
NOCACHE	▪ Advance માં એક પણ વેલ્યુ generate થશે નહીં.
ORDER	▪ Sequence નંબર Order માં જનરેટ થાય તેની ગેરેટી આપે છે આ option પેરેલલ server માં પેરેલલ મોડ માં ઉપયોગ થાય છે.
NOORDER	▪ Sequence નંબર Order માં જનરેટ થાય તેની ગેરેટી આપશે નહીં. આ option પેરેલલ server માં પેરેલલ મોડ માં ઉપયોગ થાય છે.

NEXTVAL અને CURRVAL

- Oracle માં એ pseudo column આવેલ છે: NEXTVAL અને CURRVAL
- જો sequence બનાવ્યા બાદ, SQL Statement માં CURRVAL pseudo column દ્વારા sequence ની current વેલ્યુ ને access કરી શકાય છે.
- NEXTVAL pseudo column sequence ને increment કરે છે અને નવી વેલ્યુ return કરે છે.
- Pseudo column ને નીચે મુજબ ઉપયોગ કરી શકાય:

Syntax:

- SequenceName.CURRVAL
 - Sequence ની current વેલ્યુ return કરે છે.
- SequenceName.NEXTVAL
 - Sequence ની વેલ્યુ ને increment છે અને Next વેલ્યુ return કરે છે.

Example 11 : sequence બનાવો કે જે 1 થી 99 નંબર ચડતા કરું માં જનરેટ કરે અને 99 નંબર જનરેટ થયા બાદ નંબર જનરેટ થવાના બંધ થઇ જાય.

Input:

```
Create sequence      mysequence
Start with          1
Increment by        1
Minvalue            1
Maxvalue            99
Nocycle;
```

Output:

Sequence created.

Example 12: ઉપર ના example માં બનાવેલ sequence નો ઉપયોગ કરી 'A01','A02','...A99' જીવિ sequence વેલ્યુ જનરેટ કરો.

Input:

```
Select ('A' || LTRIM (TO_CHAR(mysequence.NEXTVAL,'00'), '')) ANO FROM DUAL;
```

Output: ANO

A01

- આ જ રીતે આપણે insert અને update operation માટે query બનાવી શકીએ.

Destroying sequence

- Sequence ને નીચે મુજબ Drop કરી શકાય:

Syntax:

```
DROP SEQUENCE sequenceName;
```

Example 13: Example 11 માં બનાવેલ 'mysequence' ને drop કરો.

Input:

```
DROP SEQUENCE mysequence;
```

Output:

Sequence Dropped.

Synonyms

- Database object જેવા કે tables, indexes, sequence વગેરે માટે Alternate નામ આપવા માટે Synonyms નો ઉપયોગ થાય છે.
- Synonyms object નો ઉપયોગ વાસ્તવિક identity ને છુપાવવા માટે કરી શકાય છે.
- ઉદાહરણ તરીકે, જો કોઈ ચોક્કસ table નું નામ છુપાવવાની જરૂર હોય તો મૂળ નામ ને hide કરીને table માટે Synonyms બનાવી શકાય છે
- Synonyms નો ઉપયોગ table ના નામ ને ટુંકું કરવા માટે પણ થાય છે.
- ઉદાહરણ તરીકે, user 2 માટે user 1 દ્વારા customer table નું synonym બનાવમાં આવે છે અને તેને યોગ્ય privileges આપવામાં આવે છે.

Creating a Synonym

Syntax: CREATE SYNONYM synonymName FOR objectName;

Example 14: User 2 તરીકે user 1 દ્વારા બનાવમાં આવેલ customer table નું synonym નીચે મુજબ બનાવી શકાય છે.

Input: CREATE SYNONYM cust FOR user1.Customer;

Output: Synonym Created

Destroying a synonym:

Syntax: DROP SYNONYM synonymName;