ЎЗБЕКИСТОН РЕСПУБЛИКАСИ

ОЛИЙ ВА ЎРТА МАХСУС ТАЪЛИМ ВАЗИРЛИГИ

АНДИЖОН МАШИНАСОЗЛИК УНСТИТУТИ

“АВТОМАТИКА ВА ЭЛЕКТРОТЕХНОЛОГИЯЛАРИ”
факультети

“АХБОРОТ ТЕХНОЛОГИЯЛАРИ”
кафедраси

ORACLE маълумотлар базасини ўрганиш бўйича

УСЛУБИЙ ҚЎЛЛАНМА

Андижон – 2015

ТАСДИҚЛАНГАН.

Андижон машинасозлик институти
ўқув-услубий кенгаши йиғилишида кўриб чиқилган ва тасдиқланган.
(Кенгашнинг 2015- йил 4 08 даги 10 -сонли баённомаси)

Кенгаш раиси _____ Қ.Эрматов

МАЪҚУЛЛАНГАН.

Автоматика ва электротехнология факултети кенгаши йиғилишида
муҳокама қилинган ва маъқулланган.
(Кенгашнинг 2015- йил 06 06 даги 11 -сонли
баённомаси).

Кенгаш раиси _____ Н. Тўйчибоев

ТАВСИЯ ЭТИЛГАН.

«Ахборот технологиялари» кафедраси
йиғилишида муҳокама қилинган ва фойдаланишга тавсия этилган.
(Кафедра йиғилишининг 2015 - йил 25 04 даги
9 -сонли баённомаси)

Кафедра мудири _____ Х.Саримсақов

Тузувчилар:

Ж.Ахмадалиев – “Ахборот технологиялари“ кафедраси ассистенти.

Э.Юлдашев – “Ахборот технологиялари“ кафедраси ассистенти.

Тақризчилар:

Ё.Қурбонов – АндМИ “МИЧА“ кафедраси доценти,

М.Джалилов– ТАТУ ФФ “Компьютер тизимлари“ кафедраси доценти.

Ушбу услубий қўлланма “Ўзбекистон Республикаси Олий ва ўрта махсус таълим вазирлигининг 2008- йил 7- мартдаги 61- сонли буйруғи билан тасдиқланган “Олий таълим муассасаларида курс лойиҳасини бажариш тартиби ҳақида низом” га асосан тайёрланган бўлиб, унда курс лойиҳаси мавзулари, уларни тайёрлаш ва расмийлаштириш қоидалари келтирилган.

КИРИШ

Тараққиётнинг тез ривожланаётган хозирги даврида кўпгина сохаларни компьютерларсиз тасаввур қилиб бўлмайди. Компьютерлар халқ хўжалигининг барча сохаларига кириб келди ва кенг қўлланилмоқда. Олдинги яратилган хисоблаш машиналари фақат хисоблаш ишларини бажарган бўлса, хозирги компьютерларнинг имкониятлари жуда катта ва бир неча юзлаб амалларни бажаради.

Хозирги кунда республикамиз ташкилот ва муассаларида ахборотларни тўплаш, қайд қилиш, қайта ишлаш ва узатиш тизимларининг техник хамда дастурий таъминотлари яратилган.Барча иқтисодий ва техникавий масалаларнинг ахборот тизимини лойиҳалаш бирор алгоритмик тилда хал этилади. Ахборот тизимларини лойиҳалашнингбир қанча усуллари мавжуд бўлиб, уларнинг бир нечаси олий техника ўқув юртларининг дастурларига фаннинг асоси бўлиб киритилган.

Ахборот тизимини лойиҳалашдаги маълумотларни компьютерда тўплаш, сақлаш, қайта ишлаш ҳамда дастурий таъминотини пухта ўргатиш мақсадида ушбу услубий қўлланма тузилди. Биламизки, Oracle энг замонавий МББТ ҳисобланади. Услубий қўлланмадан фойдаланаётган мустақил ўрганувчилар назарий қисмини ўрганиб чиқишлари керак. Келтирилган буйруқлар ва функцияларни мустақил қўллай билшлари лозим. мустақил ўрганувчилар келтирилган топшириқларни бажаришлари, қўйилган саволларга жавобларни ва қилинган ишлар натижасини аълохида дафтарга қайд этиб боришлари талаб қилинади.

1 – МАВЗУ

КИРИШ. ФАННИНГ МАҚСАДИ ВА ВАЗИФАЛАРИ. АСОСИЙ ТУШУНЧА ВА ТАЪРИФЛАР. ORACLE РМББТ. ORACLE МББТ НИ ҚИСҚАЧА РИВОЖЛАНИШ ТАРИХИ. ШАХСИЙ МББТ. СЕРВЕР АСОСИДАГИ ТИЗИМЛАР. РЕЛЯЦИОН МАЪЛУМОТЛАР БАЗАСИ МОДЕЛИ. ORACLE АРХИТЕКТУРАСИ.

Режа:

1. *Маълумот базасини бошқариш тизими*
2. *Қисқача тарихий маълумот*
3. *Асосий тушунчалар*
4. *Реляцион МББТ асосий компонентaлари*
5. *Шахсий маълумотлар базасини бошқариш тизими*
6. *МББТ уч бўғинли тизими*

Маълумот базасини бошқариш тизими

Маълумотлар базасини бошқарниш тизими МББТ (СУБД, DBMS-Database Management Systems) замонавий корхоналарда муҳим компьютер воситаларидан ҳисобланади. МББТ – компьютерни операцион тизими бирга ишлайдиган ва компьютерда маълумотларни сақлаш ва чиқариб олиш учун ишлатиладиган дастурий таъминотдир. Маълумотларни ўзи ишончли ишлаб чиқилган маълумот базасида сақланади. Ҳозирги кунда асосан реляцион маълумотлар базасига асосланган тизимлар кўп ишлатилмоқда.

Унча катта бўлмаган ташкилотларда МББТ битта компьютерда юклаб ишлатиш мумкин. Бундай шахсий маълумот база тизими бир нечта фойдаланувчи ишлаганда фойдали бўлиши мумкин. Шунинг учун хозирги кунда корхона ва ташкилотларда маълумот базаси бирламчи серверда жойлаштирилади, фойдаланувчилар эса алоҳида компьютерлар билан уланади. Бу компьютерларни мижоз компьютерлари дейилади. Сервер маълумотларни марказлашган сақловчиси ҳисобланади, шунинг учун барча

фойдаланувчилар бир ахборотларни,жумладан ўзгартирилган ахборотларни кўриш имкониятига эга бўлади.

Қисқача тарихий маълумот

ORACLE корпорацияси ахборотлар бошқариш дастурий таъминотини етказиб бериш бўйича етакчи компаниялардан бири ҳисобланади. 1977й. Л.Эллисон, Б. Майнер ва Э. Оуэтс Relational Software Incorporated (RSI) фирмасини ташкил қилдилар. Шу билан ***тақсимланган маълумотлар базасини бошқариш тизими*** *(ТМББТ) ORACLE ни яратишга асос солинди. 1979й. ТМББТ ORACLE2 дастурий тизим бозорига тақдим этилди. У RSX операцион тизими билан ишлаган.*

- ✓ *1983 й ЯНГИ ORACLE 3 версияси пайдо бўлди;*
- ✓ *1984 й ORACLE 4 версияси яратилди;*
- ✓ *1984 й. ORACLE 5 версияси яратилди;*
- ✓ *1988 й. ORACLE 6 версияси яратилди;*
- ✓ *1993 й. ORACLE 7 версияси яратилди;*
- ✓ *1997 й. ORACLE 8 версияси яратилди;*
- ✓ *1999 й. ORACLE 8i версияси яратилди;*
- ✓ *2000 й. ORACLE 9 версияси яратилди;*
- ✓ *2004 й. ORACLE 10g версияси яратилди;*
- ✓ *2008 й ORACLE 11 версияси яратилди.*

Асосий тушунчалар

Жадваллар *(Tables) – Маълумотлар базасини ташкил қилувчи асосий жадваллар. Жадваллар сатр ва устунлардан ташкил топади. Битта ёки бир нечта жадваллар жадвал соҳасини (Tablespace) ичида сақланади.*

Блок *(block) – (ТМББТ) ORACLE маълумотларни сақлашни энг кичик бирлиги . У блокни ўзи (маълумотлар ёки PL/SQL –коди) ва ахборотлар сарлавҳасидан иборат. Блокни ўлчами 2 дан 16 Кбайтгача бўлади.*

Буфер *(buffer) – маълумотларни сақлаш учун ишлатиладиган бирорта оператив хотира ҳажми. Буферда ишлатишга мўлжалланган ёки яқин орада ишлатилган маълумотлар жойлашади. Кўпгина ҳолларда буферда қаттиқ дискда сақланаётган маълумотларни нусхалари жойлашади. Буфердаги маълумотлар ўзгартирилиши мумкин ва диск ёзиб қўйилиши ёки вақтинча сақловчи буферга жойлаштирилиши мумкин. Буферлар тўплами (гуруҳи) маълумотларни кэш буферни ташкил қилади. Шунингдек буферда ўзгаришлар журналидаги вақтинчалик ёзувлар сақланиши мумкин. Улар кейинчалик дискка ёзилади (ўзгаришлар журнали буфери).*

Ифлос буфер *(Dirty buffer) – маълумотлари ўзгартилган буфер. Характеристикаларни динамик жадваллари (Dynamic Performance Tables) – ORACLE нусхаси ишга тушганда автоматик равишда яратиладиган жадвал ва у шу нусхани характеристикаларини сақлаш учун ишлатилади. Улар ўз ичида боғланишлар, киритиш(чиқариш), мухит параметрларини бошланғич қийматлари ҳақидаги ахборотларни сақлайди.*

Индекс *(Index) – Индекслар устун ёки устунлар гуруҳи бўйича яратилади. Индекс маълумотларни тезкор ва самарали топиш имконини беради. Жадвалларга мурожат индексланган устун (ёки устунлар) бўйича амалга оширилади.*

Кластер *(Cluster) – Кластер бу жадваллар гуруҳ (тўплами) бўлиб биргаликда ишлатадиган умумий устунга эга бўлади ва физик нуқтаи назардан битта жадвалдай сақланади. Агар маълумотлар икки ёки бир нечта жадваллардан бир вақтда умумий устундаги маълумотлар бўйича қидиралаётган бўлса, кластерлар катта самара бериши мумкин. Жадвалларга улар кластерни қисми бўлса ҳам алоҳида мурожат қилиши мумкин.Кластерларни структураси боғланган маълумотларга бир вақтда*

мурожат қилинганда киритиш ва чиқариш харажатларини камайтириш имконини беради.

Тасавур *(View) – маълумотлар базасида мавжуд бўлиши шарт бўлмаган, лекин алоҳида фойдаланувчини сўровини бажариш вақтида яратиладиган виртуал жадвалдир. Тасавурлар билан жадваллар билан бажариладиган амалларни (сўровлар қуриш, олиб ташлаш, тиклаш) чекланмаган ҳолда бажариш мумкин.*

Синонимлар *(Synonyms) – Маълумотлар базаси объектлари учун альтернатив номлар.*

Кетма – кетликлар *(Sequences) – Маълумотларни кэш буферида сақланувчи рақамлар кетма – кетлигини яратиш учун ишлатилади.*

Маълумотлар луғати *(Data Dictionary) – Маълумотлар базасидаги ахборотларни таъминлаш учун ишлатиладиган жадваллар тўплами.*

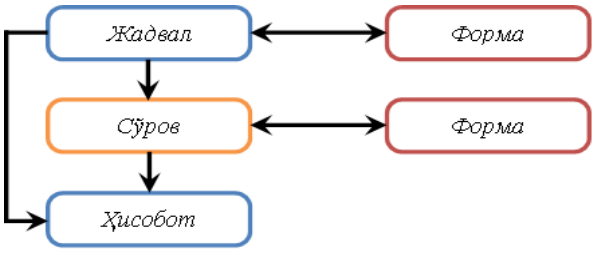
Схема *(Schema) - маълумот база объектларини мажмуаси.*

Тоза буфер *(clean buffer) – Ичидаги маълумотлар ўзгартирилмаган буфер. Ундаги маълумот ўзгармагани учун уни қаттиқ диск ёзишга зарурат йўқ.*

Реляцион МББТ асосий компонентaлари

МББТ қуйидаги компонентaлар ташкил топади:

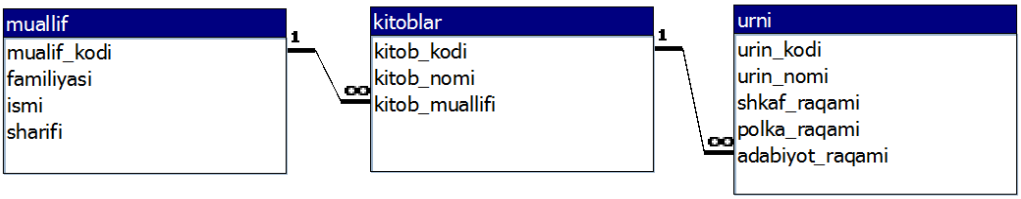
- ✓ *жадваллар;*
- ✓ *сўровлар;*
- ✓ *формалар;*
- ✓ *ҳисоботлар;*
- ✓ *бошқарувчи дастурлар.*



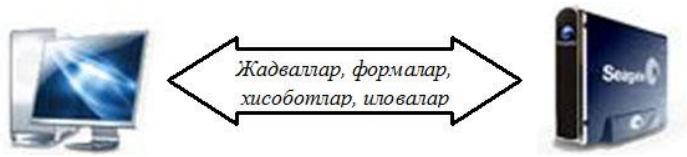
Таркибига муаллифлар, китоблар ва уларни жойлашиш жадваллари ва бу жадваллар орасидаги боғланиш ўрнатилган маълумотлар базаси яратинг.

Жадваллар қуйидаги схемалар билан берилган:

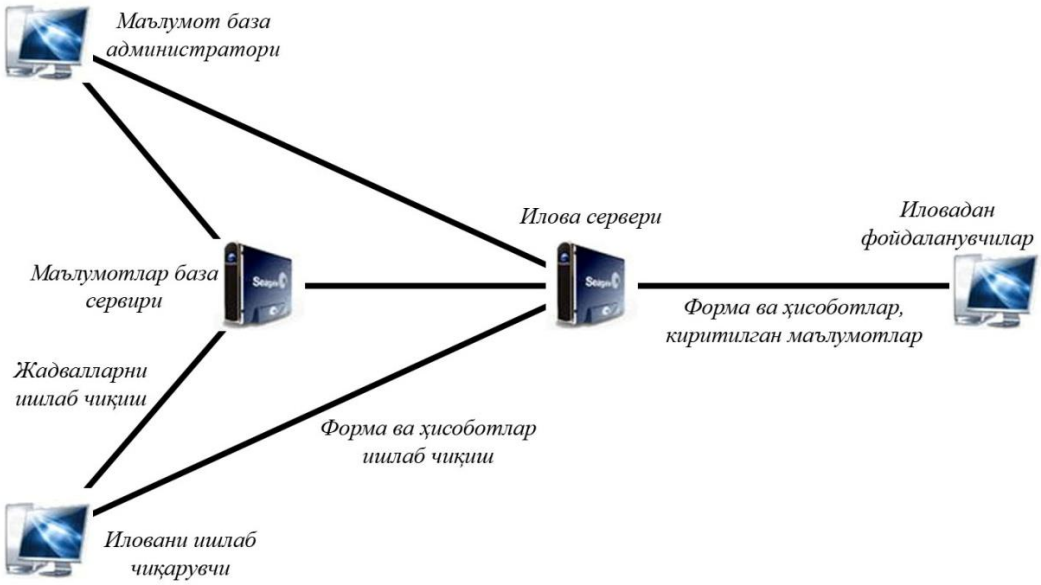
- ✓ *муаллифлар (муаллиф_коди, фамилияси, исми, шарифи) ;*
- ✓ *китоблар (китоб_коди, номи, китоб_муаллифи);*
- ✓ *ўрни (ўрин_коди, китоб_номи, шкаф_№, полка_№).*



Шахсий маълумотлар базасини бошқариш тизими



МББТ уч бўғинли тизими



Назорат саволлари:

1. *Асосий тушунчаларни санаб беринг?*
2. *Кластер нима?, Тушунча нима?*
3. *Реляцион МББТ асосий компонентaлари нималардан иборат?*
4. *Oracle МББТ қачон ва кимлар томонидан яратилган.*

2 – МАВЗУ

ORACLE НИ АСОСИЙ ОБЪЕКТЛАРИ. ORACLE ТИЛИНИНГ МАЪЛУМОТЛАР БИЛАН ИШЛАШ (МАНИПУЛЯЦИЯЛАШ) ВОСИТАЛАРИ. СЎРОВ СТРУКТУРАСИ. ОДДИЙ СЎРОВЛАР.ТАНЛАБ ОЛИШ МЕЗОНИНИ ТУЗИШ. ТАРТИБЛАШ.

Режа:

1. SQL стандартлари
2. ORACLE асосий объектлари
3. SQL тилини маълумотлар билан манипуляциялаш воситалари
4. SELECT оператори
   - ✓ Содда сўровлар
   - ✓ Тўла номни ишлатиш
   - ✓ Distinct параметридан фойдаланиш
   - ✓ Танлаш мезонлари
   - ✓ Шарт ифодалари
   - ✓ Ажратиш мезони аниқлаш воситалари

SQL стандартлари

SQL замонавий МББТ учун маълумотларни қайта ишлашни тил воситаларини стандарти ҳисобланади. Ҳозирги кунда Америка миллий стандартлар институти (ANSI) томонидан қабул қилинган (SQL-86) SQL тилини қайта кўриб чикилган стандарти мавжуд: SQL89, SQL92 (ISO/IES 9075:1992), SQL2003 (ISO/IES 9075:2003). Бунда ҳар бир стандарт олдингисини аниқлаштиради ва кенгайтиради. Умуман олганда, стандартларга бўйсиниш мажбур бўлиши шарт бўлмаса ҳам, дастурий таъминот яратувчилар (МББТ) SQL89 ва SQL92 талаблари ни бажаришга ҳаракат қиладилар.

SQL операторларини 3 та гуруҳга бўлиш мумкин:

✓ *DML (Data Manipulation Language) – маълумотлар билан манипуляциялаш операторлари;*

✓ *DDL(Data Definition Language) – маълумотларни аниқлаш операторлари, транзакцияларни бошқариш операторлари гуруҳи ва мурожатларни тақдим қилиш операторилари гуруҳи;*

✓ *DDL операторлар гуруҳи маълумот база объектларини яратиш, такомиллаштириш ва олиб ташлаш учун мўлжалланган.*

ORACLE асосий объектлари

ORACLE асосий объектларига қуйидагилар киради: жадваллар, тасавурлар ва фойдаланувчи.

Фойдаланувчи (USER) – объект бўлиб, у ORACLE ни бошқа объектларини яратиш ва ишлатишга, ҳамда сервер функцияларини бажаришни сўраш имкониятига эга. Бундай функциялар сонига сессияларни ташкил қилиш, маълумот базасини холатини ўзгартириш ва бошқалар киради. ORACLE фойдаланувчиси билан схема (SCHEMA) боғланган.

Схема фойдаланувига тегишли бўлган маълумотлар базаси объектларини мантиқий тўплами (жадваллар, кетма кетликлар, сақланувчи дастурлар). Схема фақат битта эга фойдаланувчига тегиш бўлиб, фақат у бу объектларни яратиш ва олиб ташлашга жавобгардир. Схема қуйидаги объектларни сақлаши мумкин: кластерлар, маълумот база боғланишлари, триггерлар, ташқи процедура библиотекаси, индекслар, пакетлар,кетма кетликлар, сақланувчи функция ва процедуралар, синонимлар, жадваллар, тасавурлар.

Жадвалар (TABLE) реляцион маълумотлар базасини асосини ташкил қилади. Реляцион маълумотлар базасида барча ахборотлар жадвалларда сақланади. Жадвалларни тўла номи схема номи ва жадвал номидан ташкил топади.

Роль (ROLE) – Фойдаланувчилар ёки бошқа ролларга тақдим этиладиган имтиёзларни мажмуаси. Жадвал соҳаси (TABLESPACE) - маълумот базасини номланган қисми бўлиб, жадваллар, индекслар ва бошқа объектлар учун хотирани тақсимлаш учун ишлатилади. Маълумотларни қайта ишлаш алгоритмларин дастурлаш учун, маълумот базасини яхлитлигини динамик таъминлаш механизмини амалга ошириш учун ORACLE қуйидаги объектларни ишлатади: Процедура (Procedure), функция (FUNCTION), пакет (PACKAGE), триггер (TRIGGER) каби объектларни ишлатилади.

SQL тилини маълумотлар билан манипуляциялаш воситалари

SQL тилида маълумотлар билан манипуляция амаллари учун тўртда калит сўз ишлатилади: SELECT, INSERT, UPDATE ва DELETE. SELECT операторли ифодалар алоҳида ўрин эгаллайди, чунки у маълумотлар ажратиб олиш учун мўлжалланган ва бу фойдаланувчилар ечадиган масалаларни кўп қисмини ташкил қилади.

SELECT оператори

SELECT хизматчи сўзи маълумот базасидан ахборотни танлаб олиш оператори ёзилганини англатади. SELECT сўзидан кейин бир биридан вергул билан ажратилиб сўралаетган майдон номлари (атрибутлар рўйхати) ёзилади. SELECT сўров операторини зарур хизматчи сўзи FROM (ундан, дан) хисобланади. FROM сўзидан кейин ахборот олинаётган жадвал номлари бир биридан вергул билан ажратилиб ёзилади.

SELECT оператори кўрсатилган ажратиш мезонига мос равишда бир ёки бир неча жадвалдан атрибутларни ажратиш учун ишлатилади. SELECT (танлаш) SQL тилининг энг муҳим ва кўп ишлатиладиган оператори ҳисобланади. У маълумотлар базаси жадвалидан ахборотларни танлаб олиш учун мўлжалланган.

SELECT оператори содда ҳолда қуйидаги кўринишда ёзилади:

```
SELECT [DISTINCT] <атрибутлар руйхати>
FROM <жадваллар руйхати>
[WHERE <танлаш шарти>]
[ORDER BY <атрибутлар руйхати>]
[GROUP BY <атрибутлар руйхати>]
[HAVING <шарт>]
[UNION <SELECT операторли ифода>]
```

Бу ерда квадрат қавсларда ёзилган элементлар хар доим хам ёзилиши шарт эмас.

Содда сўровлар

SQL тилида барча майдонларни (атрибутларни) чиқариш учун «» юлдузча белгиси ишлатилади. Мисол: Қуйидаги операторлар билан жадвал яратилган ва тўлдирилган бўлсин.*

```
CREATE TABLE Tab1 (At1 CHAR(3), At2 NUMBER);
INSERT INTO Tab1 VALUES (‘A’, 1);
INSERT INTO Tab1 VALUES (‘B’, 2);
INSERT INTO Tab1 VALUES (‘C’, 2);
```

Tab1 жадвалидан барча маълумотларни чиқариш бажарувчи сўров кўриниши:

```
SQL> SELECT * FROM Tab1;
```

Операторни бажариш натижаси:

At1	At2
A	1
B	2
C	2

Тўла номни ишлатиш

Агар маълумотлар фойдаланувчи схемасидан фарқли бўлган схемадан чиқарилса, сўровда жадвални тўла номи кўрсатилади. Масалан, u1 фойдаланувчи u2 фойдаланувчини Tab1 жадвалидан танлашни бажарсин. u2 фойдаланувчини схемасини Tab1 жадвалидан барча маълумотларни чиқариш бажарувчи сўров қуйидагича берилади:

```
SQL> SELECT * FROM u2.Tab1;
```

Distinct параметридан фойдаланиш

Мисол:

```
SQL> SELECT DISTINCT At1 FROM Tab1;
```

Натижа:

At1
A
B
C

Мисол:

```
SQL> SELECT DISTINCT At2 FROM Tab1;
```

Натижа:

At2
1
2

Танлаш мезонлари

SELECT операторида WHERE параметрини (операторини) ишлатиб, биз рост ва ёлғон қийматларни қабул қилувчи шартли ифодаларни ёзишимиз мумкин. WHERE параметрида ёзилган шарт, SELECT оператори ёрдамида

кўрсатилган жадваллардан, қайси сатрлар танлаб олиниши кераклигини аниқлаб беради. Сўров натижаси бўлган жадвалга, WHERE параметрида шарти кўрсатилган ифодани рост бўлган қийматларини қаноатлантирувчи сатрлар киритилади.

Масалан, Tab1 жадвалидаги 'A' сатрли ёзувларни At2 майдони қийматини чиқаринг:

```
SQL> SELECT At2 FROM Tab1 WHERE At1='A';
```

Натижа:

At2
1

Шарт ифодалари

WHERE параметрида бериладиган шартларда мунособат (солиштириш) амаллари = (тенг), > (катта), < (кичик), >= (катта ёки тенг), <= (кичик ёки тенг), <> (тенг эмас), шунингдек мантикий (логик) AND, OR ва NOT амаллар ишлатилади.

Масалан,

```
SQL> SELECT At2 FROM Tab1 WHERE At2 <> 1;
```

Натижа:

At1	At2
B	2
C	2

Ажратиш мезони аниқлаш воситалари

SQL стандартида маълумотлар базасида аниқланмаган қийматларни ишлатишга рухсат беради. Шунинг учун танлаш шартлари буль мантиқида эмас, уч қийматли мантиқда (TRUE, FALSE, UNKNOWN (НОМАЪЛУМ)).

Буль амаллари AND, OR, UNKNOWN уч қийматли мантиқда қуйидагича ишлайди:

Амаллар	**Натижа**
TRUE AND UNKNOWN	UNKNOWN
FALSE AND UNKNOWN	FALSE
UNKNOWN AND UNKNOWN	UNKNOWN
TRUE OR UNKNOWN	TRUE
FALSE OR UNKNOWN	UNKNOWN
UNKNOWN OR UNKNOWN	UNKNOWN
NOT UNKNOWN	UNKNOWN

Назарий саволлар

1. Шарт орқали сўровлар қандай амалга оширилади?

2. Бошқа фойдаланувчига тегишли бўлган жадваллар ни кўришда қандай сўров ёзилади?

3. Жадвални тўлтириш сўровини қандай ёзилади

4. DISTINCT ҳизматчи сўзи қандай иш бажаради?

3 – МАВЗУ

ORACLE ДА МАЪЛУМОТЛАРНИ ТАВСИФЛАШ ТИЛИ. ORACLE ДА МАЪЛУМОТ ТОИФАЛАРИ ВА ОБЪЕКТЛАРИ. СИМВОЛЛИ САТР. СОНЛИ ТОИФАЛАР. ROWID ТОИФАСИ. БИТЛАР САТРИ. САНА ВА ВАҚТ ТОИФАСИ. LOB – ОБЪЕКТЛАРИ.

Режа:

1. *DDL ни асосий функциялари*
2. *Oracle да маълумот тоифалари*
3. *Сонли тоифалар*
4. *ROWID тоифаси*
5. *Битли сатрлар*
6. *Сана ва вақт тоифалари*
7. *LOB – объектлари*

DDL ни асосий функциялари

- ✓ *Маълумот тоифаларини идентификациялаш;*
- ✓ *Ҳар хил тоифа маълумотларига ягона ном тайинлаш;*
- ✓ *Маълумот база объектларини структурасини спецификациялаш;*
- ✓ *Калитларни спецификациялаш;*

Маълумотларни тавсифлаш тилини қўшимча имкониятлар сонига қўшиш мумкин:

- ✓ *Маълумот элементларини хусусий характеристикаларин аниқлаш;*
- ✓ *Яхлитликни чеклашни аниқлаш;*
- ✓ *Маълумотларни сақлашни физик босқичи элементларини тавсифлаш.*

Маълумот элементларини хусусий характеристикаларига одатда символли сатрларни узунлигини аниқлаш, сонли маълумотларни масштаб, аниқлиги ва ҳоказо.

***Эслатма**: Эътибор беринг, маълумотларни сақлаш формати, фойдаланувчи олаётган маълумот форматига мос келиши мажбур эмас. Oracle сервери маълумотларни қийматини қайта форматлаш имкониятини таъминлайди.*

Яхлитликни чеклаш кўрсатмаси маълумотларни киритиш ишончлигини ошириш учун ишлатилади. Яхши структуралашган маълумотларни автоматик назорат маълумот базасидаги ахборотларни ишончлигини оширади.

Маълумотларни сақлашни физик босқичини тавсифлаш воситалари баъзан хотирани тавсифлашни махсус қисм тили сифатида ажратилади. Одатда МББТи ўз таркибига маълумотларни сақлаш хусусиятларини тавсифлаш учун воситаларни олади.

Oracle да маълумот тоифалари

SQL нинг 1992 йилги стандартларида санаб ўтилган барча тоифалар Oracle томонидан таъминланади.

Символли сатрлар. CHARACTER тоифаси фиксирланган узунликдаги сатрларни сақлаш учун ишлатилади. Сатрларни сақлаш учун узунлик параметрида кўрсатилган соҳа резерв(заҳира) қилиб қўйилади. Зарур бўлганда қисқа сатр пробеллар билан тўлдирилади.

Синтаксиси: **CHARACTER**[(узунлик)], **CHAR**[(узунлик)]

Агар сатр узунлиги ошкор ҳолда кўрсатилмаса, у 1 тенг деб ҳисобланади. Узунлик параметрини максимал қиймати- 256 символдан иборат.

Мисол: str1 **CHAR**(10), Str2 **CHARACTER**

VARCHAR тоифаси ўзгарувчан узунликдаги сатрларни сақлаш учун ишлатилади. Сатрларни сақлаш учун реал зарур бўлган соҳа резерв(заҳира) қилиб олинади.

Синтаксиси:
VARCHAR[(узунлик)], **CHAR VARYING** [(узунлик)], **CHARACTER VARYING** [(узунлик)]

Агар сатр узунлиги ошкор берилмаган бўлса, у 1 тенг деб ҳисобланади. Узунлик параметрини максимал қиймати – 4000 символдан иборат.

Мисол: varstr1 **VARCHAR**(10)

VARCHAR2 тоифаси [фақат ORACLE учун]. Ўзгарувчан узунликдаги сатрларни сақлаш учун ишлатилади. Сатрларни сақлаш учун реал зарур бўлган соҳа резерв(заҳира) қилиб олинади. VARCHAR2 тоифани киритишга сабаб фирма ORACLE кейин версияларида бу тоифани ўзгармаслигини таъкидлайди, ва ҳоланки VARCHAR тоифаси SQL стандартлар талабларига мос келади.

Синтаксиси: **VARCHAR2**(узунлик)

Сатр узунлиги ошкор кўрсатилиши шарт. Узунлик параметрини минимал қиймати – 1 символ. Узунлик параметрини максимал қиймати – 4000 символ.

Мисол: str1ora **VARCHAR2**(10)

Бу икки тоифа ўртасида фарқ бор: VARCHAR ўзгарувчан узунликдаги сатрларни сақлаш учун ишлатилади. Сатрларни сақлаш учун реал зарур бўлган соҳа резерв (заҳира) қилиб олинади.

CHAR тоифадаги маълумотлар фиксирланган узунликка эга ва 255 тагача символдан ташкил топиши мумкин. Ваҳоланки, VARCHAR2 ўзгарувчан узунликка эга ва 4000 тагача символга эга бўлиши мумкин. Ихтиёрий CHAR(255) тоифали устун қиймати 255 байт диск соҳасини

эгаллайди. Бу фарқни жадвал структурасини лойихалашда, маълумотларни сақлаш учун жой ажратишда ва сатрларни SQL – ифодалар солиштиришда ҳисобга олиш керак. Агар иккита сатр ҳар хил узунликка эга бўлса, уларни солиштиришда қисқароқ сатр иккинчи узун сатр узунлигича пробел билан тўлдирилади. Миллий алфавитни ишлатадиган символли маълумотларни сақлаш учун NCHAR ва NVARCHAR2 тоифалари мўлжалланган.

Сонли тоифалар

INTEGER тоифаси. INTEGER тоифаси – 2^{31} дан 2^{31} гача диапазондаги бутун сонларни тасвирлаш учун ишлатилади.

Синтаксиси: **INTEGER, INT** Мисол: varint1 **INTEGER**, varint2 **INT**

NUMBER тоифаси [фақат Oracle учун].NUMBER тоифаси берилган аниқликда сонларни тасвирлаш учун ишлатилади.

Синтаксиси: **NUMBER** [(аниқлик [,масштаб])]

Агар аниқлик параметрини қиймати ошкор ҳолда кўрсатилмаса, у 38 тенг деб ҳисобланади. Масштаб параметрини қиймати сукут билан 0 тенг деб олинади. Аниқлик параметрини қиймати 1дан 38 гача ўзгариши мумкин, масштаб парамеирини қиймати – 84 дан 127 гача ўзгариши мумкин. Масштабни манфий қийматларини ишлатиш ўнли нуқтани юқори разрядлар томон силжишини англатади. Масалан, NUMBER (7, -3) минглар хонасигача яхлитлашни англатади.

Мисол: varcounter **NUMBER**

Қуйидаги жадвалда 123456.789 сонини ҳар хил аниқлик ва масштабда тасвирланиши келтирилган.

Аниқланиши	**Амаллар бажаришдаги кўриниши**

Number	123456
Number(7,1)	123456.7
Number(5,2)	Маълумотларда ҳато
Number(9,3)	123456.789
Number(7,-2)	123500

Бошқа МББТ билан мос таъминлаш учун Oracle DECIMAL, DOUBLE_PRECISION, NUMERIC, DEC ва REAL маълумот тоифаларини ҳам ишлатади.(таъминлайди). Сонли тоифаларни барчаси реал равишда Oracle ички бир хил форматида сақланади. DECIMAL ва NUMERIC тоифалари NUMBER тоифасига тўла эквивалент.

Синтаксиси:
DECIMAL [(аниқлик [,масштаб])], **DEC** [(аниқлик [,масштаб])]
NUMERIC [(аниқлик [,масштаб])]
Мисол:
vardecl **DEC**,vardec2 **DEC**(5),vardec3 **DECIMAL**(8,3),varnum **NUMERIC**

ROWID тоифаси

ROWID – махсус маълумот тоифаси бўлиб, жадвалдаги ёзувга кўрсаткичларни тасвирлаш(қўйиш) учун хизмат қилади. Жадвалда сатр яратилганда унга дарҳол ROWID тайинланади ва у маълумотлар қайта ташкил қилингунга ёки уни олиб ташлангунга қадар ўзгармасдан қолади. ROWID ишлатиш жадвалдаги сатрга мурожаатни энг тезкор усулидир.

Битли сатрлар

RAW тоифаси [фақат Oracle учун] – Ўзгарувчан узунликдаги иккилик сатрларни сақлаш учун ишлатилади. RAW тоифани CHAR, VARCHAR2 тоифадан фарқи шундан иборатки, Oracle символли тоифалар учун клиент билан сервер орасида уларни узатаётганда маълумотларни автоматик

алмаштиради. Oracle RAW тоифасидаги маълумотларни ўнолтилик кўринишда чиқаради.

Синтаксиси: **RAW** [(узунлик)]

Узунлик параметри байтларда ўлчанади. Узунлик параметри максималь қиймати – 2000 байт.

Мисол: bitarray1 **RAW**(10)

LONG RAW тоифаси[фақат Oracle учун]. Ўзгарувчи узунликдаги катта битли сатрларни сақлаш учун ишлатилади.

Синтаксиси: **LONG RAW** [(узунлик)]

Узунлик параметри байтларда ўлчанади. Агар сатр узунлиги ошкор ҳолда берилган бўлмаса, у 2 мегабайтга тенг деб ҳисобланади. Узунлик параметрини максимал қиймати – 2гигабайт символдан иборат. LONG RAW тоифадаги ўзгарувчилар учун индексни қуриш мумкин эмас.

Мисол: bitarrayl **RAW**(10)

Сана ва вақт тоифалари

DATE тоифаси [фақат Oracle учун] – бу тоифа сана ва вақт сақлаш учун ишлатилади. Эрамиздан аввалги 1 январь 4712 дан то 31 декабрь 4712 эрамизгача саналарга рухсат берилади. SQL ва PL/SQL да DATE (сана) тоифаларини қийматини яратиш учун одатда киритилган функциялар TO_DATE (“санани символли сатри”,”сана_формати”) ишлатилади. Вақтни белгиламасдан сана аниқланса, сукут билан ярим кечаси вақти қабул қилинади. SYSDATE функцияси жорий сана ва вақтни чиқаради. Функцияни қиймати Oracle сервери ишлаётган компьютер операцион тизими воситалари билан аниқланади.

Синтаксиси: **DATE**	Мисол: birthday **DATE**

Сана ва вақтни сақлаш учун махсус тоифани (борлиги) мавжудлиги сана ва вақт махсус арифметика амалларини бажариш имконини таъминлайди. DATE тоифасидаги ўзгарувчига бутун сонни қўшиш Oracle томонидан етарли кечки санани аниқлаш деб, айириш эса олдинги сана аниқлаш деб талқин қилинади.

Мисоллар:

```
SQL> SELECT SYSDATE FROM dual;
SYSDATE
16-09-2011
SQL> SELECT SYSDATE+10 "sysd+10" FROM dual;
sysd+10
26-09-2011
SQL> SELECT SYSDATE-10 "sysd-10" FROM dual;
sysd-10
06-09-2011
```

LOB – объектлари

BLOB тоифаси [фақат Oracle учун] ўлчами 4 гигабайт бўлган иккилик маълумотларни сақлаш учун ишлатилади. Катта иккилик объектлари билан ишлаш учун DBMS_LOB пакети ишлатилади.

Синтаксиси: **BLOB**	Мисол: doc **BLOB**

CLOB тоифаси [фақат Oracle учун] бир байтли кодировкани ишлатадиган ўлчами 4 гигабайтгача ўзгарувчан узунликдаги символли маълумотларни сақлаш учун ишлатилади.

Синтаксиси: **CLOB**	Мисол: cdoc **CLOB**

NCLOB тоифаси [фақат Oracle учун] битта ёки кўп байтли кодировкани ишлатадиган, ўлчами 4 гигабайтгача бўлган символли маълумотларни сақлаш учун ишлатилади.

Синтаксиси: **NCLOB**	Мисол: ncdoc **NCLOB**

BFILE тоифаси [фақат Oracle учун] МББТга нисбатан ташқи файлларда жойлашган, иккилик маълумотлар кўрсаткичларини сақлаш учун ишлатилади. Файлларни ўзи файл тизимида сақланади

Синтаксиси: **BFILE**	Мисол: File_ doc **BFILE**

Назарий саволлар

1. *LOB – объектларни санаб беринг.*
2. *Сана ва вақт тоифалари ишлатишни тушунтириб беринг.*
3. *Сонли тоифалар ҳақида маълумот беринг.*

4 – МАВЗУ

ЖАДВАЛЛАР. ТАСАВУРЛАР. SQL НИ МАХСУС ПРЕДИКАТЛАРИ. IN, BEETWEEN, LIKE ВА БОШҚАЛАР.

Режа:

1. *CREATE TABLE операторини қисқартирилган конструкциялари*
2. *ALTER TABLE оператори ва DROP TABLE оператори*
3. *Тасаввурлар, тасаввурни олиб ташлаш ва сатрларни қўшиш амали*
4. *DELETE Сатрларни олиб ташлаш амали*
5. *UPDATE сатрларни такомиллаштириш амали*
6. *Махсус предикатлар (шарт параметрлари)*
7. *Between параметри ва NULL предикати*
8. *Қисм сўровлар*

CREATE TABLE операторини қисқартирилган конструкциялари

CREATE TABLE [схема_номи.]жадвал_номи
((жадвал яхлитлигини чеклаш|устун_номи Устун маълумотларини тоифаси [DEFAULT ифода][устун яхлитлигини чеклаш...]},...)[{ **CLUSTER** кластер_номи (устун_номи,[...])|(**PCTFREE** бутун|**PCTUSED** бутун|**INITRANS** бутун|**MAXTRANS**бутун|**TABLESPACE** жадвал_соҳасини_номи |**STORAGE** хотира_ўлчами|{**RECOVERABLE**|**UNRECOVERABLE**}}...][**PARALLEL** паралел_қайта_ишлов_имконияти][{**ENABLE** яхлитликни_текширувчи_чекловлар|**DISABLE** яхлитликни_эътиборга_олмайдиган_чекловлар}...][**AS** сўров][**CACHE**|**NOCACHE**]

Мисол 1: u1 фойдаланувчи схемасида учта At1, At2, At3 атрибутли Tab1 жадвални яратишга доир мисол кўриб чиқамиз. Қуйидаги чекланишларни ҳисобга оламиз:

pk_Tab1_At1 - At1 атрибут бирламчи калит эканлигини кўрсатади;

nn_Tab1_At2 - At2 атрибутга аниқмас қийматлар киритиш мумкин эмаслигини кўрсатади;

At3 – атрибутни қиймати сукут билан жорий санани англатади.

```
SQL> CREATE TABLE ul.Tab1 (At1 NUMBER CONSTRAINT pk_Tab1_At1
PRIMARY KEY, At2 NUMBER CONSTRAINT nn_Tab1_ At2 NOT NULL,
At3 DATE DEFAULT SYSDATE);
TABLE created.
```

Бу мисолда аниқ жадвал соҳасида ва аниқ сақлаш параметрлари билан жойлаштирилган жадвални яратиш протоколи келтирилган. Мисол 2: Қуйидаги иккита At1 ва At2 атрибутли Tab2 яратишни намойиш қилади. Жадвал app_data жадвал соҳасига (олдиндан яратилган бўлиши керак) бирламчи калит билан боғланган, index_data жадвал соҳасига индекс билан жойлашган, Жадвал учун 100 килобайт бошланғич экстент резервланади (ажратилади) ва 50 килобайтда орттирма экстент аниқланган.

```
SQL> CREATE TABLE Tab2(At1 NUMBER CONSTRAINT pk_Tab1_At1
PRIMARY KEY USING INDEX TABLESPACE index_data, At2 NUMBER)
TABLESPACE app_data STORAGE (INITIAL 100K NEXT 50K);
TABLE created.
```

Аниқ жадвал соҳасида ва аниқ сақлаш параметлари билан жойлаштирилган жадвални яратиш протоколи.

ALTER TABLE оператори

ALTER TABLE оператори ёрдамида бир ёки бир неча янги устун қўшиш, яхлитликни чеклаш, мавжуд устунларни тавсифини (маълумот тоифасини, унинг узунлигини, сукутли қиймат ёки NOT NULL яхлитликни чеклаш) такомиллаштириш, сақлаш ва транзакция параметларини (PCTFREE, PCTUSED, INITRANS, MAXTRANS, NEXT, PCTINCREASE)

такомиллаштириш мумкин. Жадвалларни ўзгартириш ва айниқса яхлитликни чеклашни ёқиш / ўчиришни ўзига хос хусусиятлари бор.

Операторни синтаксиси:

ALTER TABLE «жадвал_номи» **ADD** («устун_номи» «маълумот_тоифаси» «ўлчами»);

Бу команда бўйича мавжуд жадвал сатрларига янги устун қўшиладива унга NULL қиймати киритилади. Қўшилган устун жадвалда охирги ҳисобланади. Бир нечта устун ҳам қўшиш мумкин. Бу ҳолда улар вергул билан ажратилади. Устунни тавсифини ўзгартириш имконияти бор. Кўпинча бу устун узунлигини ўзгартириш, уларни қийматларига чекланишларни қўшиш ёки олиб ташлаш. Бу ҳолда команда синтаксиси қуйидагича бўлади:

ALTER TABLE «жадвал номи» **MODIFY** «устун номи» «маълумот тоифаси» «ўлчами аниқлик»;
Мисол: **ALTER TABLE** Tab1 **ADD** (At3 **NUMBER**);

DROP TABLE оператори

Маълумотлар базасидан жадвални олиб ташлаш учун (ундаги маълумотлар билан) DROP TABLE оператори ишлатилади. ORACLE да жадвални олиб ташлаш опратори куйидаги синтаксисига эга:

DROP TABLE [схема_номи.]жадвал_номи [**CASCADE CONSTRAINTS**]

Мисол: Tab1 жадвални олиб ташлаш учун

SQL> **DROP TABLE** Tab1;
TABLE dropped;

Тасаввурлар

Тасаввур – бир ёки бир неча жадваллардан ёки тасаввурлардан олинадиган, сервер томонидан динамик равишда таъминланадиган номланаган танлашдир. Oracle да тасаввурни аниқлаш операторини синтаксиси:

CREATE [**OR REPLACE**] [{**FORCE**|**NO FORCE**}] **VIEW**[схема_номи.] тасаввур_номи [(алътернатив_ном [альтернатив_ном...])] **AS** сўров **WITH** {**READ ONLY**|**CHECK OPTION**[**CONSTRAINT** яхлитликни_чеклаш]}

OR REPLACE калит сўз эски тасаввурни янги билан мажбурий алмаштиришни кераклигини кўрсатади.

FORCE | NO FORCE (сукут билан олинади) асосий жадвалларни ва тасаввурларни мавжудлиги ва тасаввур яратувчи фойдаланувчида асосий жадвалларга мурожат учун имтиёзлар бўлишлигини мажбурийлигини кўрсатади. Сўров параметри ихтиёрий тўғри сўровни англатади. Бу сўровларда ORDER BY калит сўз ёки FOR UPDATE конструкция бўлмаслиги керак.

WITH READ ONLY калит сўз асосий (базовий) жадваллар учун кўрсатилган тасаввур билан маълумотларни такомиллаштиришни таъқиқлашни кўрсатади. CONSTRAINT калит сўзи текшириш учун ишлатиладиган чеклаш номини аниқлайди. Фақат ўқиш учун чекланишли тасаввурни яратишга доир мисол:

SQL>CONNECT ul/ulpsvf@educ;
Connected.
SQL>**CREATE TABLE** Tab2(At1 **NUMBER**, At2 **NUMBER**);
TABLE created.
SQL>**CREATE VIEW** Vul AS 2 **SELECT** * **FROM** Tab2 **WITH READ ONLY**;
View created.
SQL>**GRANT INSERT ON** Tab2 **TO** u2;

```
Grant succeeded.
SQL>GRANT INSERT ON Vul TO u2;
Grant succeeded.
SQL>CONNECT u2/u2psw@educ;
Connected.
SQL>INSERT INTO ul.Vul VALUES(1,2);
INSERT INTO ul.VUl VALUES (1,2) * ' ERROR at line 1: ORA-01732: data
manipulAT1on operAT1on not legal on this view
SQL> INSERT INTO ul.Tab2 VALUES(1,2);
```

Фақат ўқиш учун чекланишли тасаввурни яратиш протоколи.

Тасаввурни олиб ташлаш

Тасаввурни олиб ташлаш учун DROP VIEW командаси ишлатилади.Тасаввурни олиб ташлаш учун уни эгаси бўлиши керак ёки DROP ANY VIEW имтиёзига эга бўлиш керак. Командани синтаксиси қуйидагича:

```
DROP VIEW [схема_номи.]тасаввур_номи
```

Тасаввурни олиб ташлашганда, унга мурожат объектлар олиб ташланмайди, улар хақиқий бўлмай қолади.Олиб ташланаётган тасаввурларга бериладиган имтиёзлар бекор қилинади. Мисол: vu3 Тасаввурни олиб ташлашга мисол.

```
SQL>DROP VIEW vu3;
View droped.
```

Сатрларни қўшиш амали

Жадвалга ёки асосий жадвал тасаввурларига сатрлар қўшиш учун INSERT командаси ишлатилади.

Унинг синтаксиси:

```
INSERT INTO {[схема_номи.]{жадвал_номи|тасаввур_номи}
[@МБ_боғланиш_номи]|(қисм_сўров_1)}[(устун_номи[,устун_номи...])]
{VALUES(ифода[,ифода]...)}|қисм_сўров_2]
```

INSERT да ёзилган жумлалар кўрсатилган кетма кетликда ёзилиши керак. Тасаввурни ишлатиб сатр қўшилса, сатр тасаввурни асосий (базовий) жадвалига қўшилади. Мажбурий бўлмаган параметр схема номи Oracle мос объекти жойлашган схема номи аниқлаштиради. Сукут билан амални бажараётган фойдаланувчи схемаси ишлатилади. @МБ боғланиш_номи параметри узоқлашган маълумот базаси билан боғланиш номини ўрнатади. Агар узоқлашган маълумот базаси билан боғланиш номи кўрсатилмаса, унда Oraclени мос объекти асосий маълумот базасида жойлашган деб ҳисобланади. қисм_сўров_1 параметри тасаввурга деб ҳисобланадиган қисм сўровни беради. Ифода параметри ҳисобланувчи ифодага алмаштирилади. Одатда у INTO калит сўзидан кейин қийматлари кўрсатилган, жадвал, тасаввур ва сурат устунларидаги маълумотларга асосланади. қисм_сўров_2 параметри киритилаётган сатрлар тўпламини яратадиган қисм сўровни билдиради. Tab1 ва Tab2 жадвалга сатрларни қўшиш операторини ишлатишни кўриб чиқамиз. Tab1 ва Tab2 жадвал қуйидаги командалар билан яратилган ва тўлдирилган .

```
SQL>CREATE TABLE Tab1 (At1 CHAR(3), At2 NUMBER);
SQL>CREATE TABLE Tab2 (At1 NUMBER);
SQL>INSERT INTO Tab1 VALUES('A',1);
```

Листинг. Қийматлар рўйхатини ошкор кўрсатиб сатр қўшиш.

```
SQL>INSERT INTO Tab1 VALUES('B1', NULL);
```

Листинг. Атрибут қийматлардан бирига аниқмас қийматли сатрни қўшиш.

DELETE Сатрларни олиб ташлаш амали

DELETE амали жадваллардан ва тасввурларни базаси жадвалидан сатрларни олиб ташлаш учун ишлатилади.

Синтаксиси:

DELETE [**FROM**] {[схема_номи.]{жадвал_номи|тасаввур_номи}[@МБ боғланиши_номи]|қисмсўров}[альтернатив_ном][**WHERE** шарт]

DELETE да ёзилган жумлалар кўрсатилган кетма кетликда ёзилиши керак. Бу ерда WHERE параметри берилмаса жадвалдан барча сатрлар олиб ташланади. Агарда WHERE калит сўз ишлатилса, шарт бажариладиган сатрларгина олиб ташланади.

```
SQL>DELETE FROM Tab1 WHERE At1 < = 100
1 row deleted.
SQL>DELETE FROM Tab1;
1 row deleted.
```

Листинг. Танлаш мезонини ишлатиб сатрларни олиб ташлаш ва шартсиз сатрлар олиб ташлаш.

UPDATE сатрларни такомиллаштириш амали

UPDATE амали жадвалдаги, тасаввурни база(асосий) жадвалидаги (ёки расмдаги)сатрларни такомиллаштиришни амалга оширади.

Синтаксиси:

UPDATE {[схема_номи.]{жадвал_номи|тасаввур_номи\расм_номи}[@МБ боғланиш_номи]|қисм сўров_1}[альтернатив_ном] **SET** {[(устун_номи [, устун_номи...])=(қисмсўров_2)]\устун_номи={ифода\қисмсўров_3}}… [**WHERE** условие]

UPDATE да ёзилган жумлалар кўрсатилган кетма кетликда ёзилиши керак.

Мисол:

```
SQL> SELECT * FROM Tab1;
```

Натижа:

AT1	AT2
1	AAA
2	BBB

Мисол:

```
SQL> UPDATE Tab1 SET At2 = 'CCC' WHERE At1 = 1;
1 row updated. SQL> SELECT * FROM Tab1;
```

Натижа:

AT1	AT2
1	CCC
2	BBB

Мисол:

```
SQL> UPDATE Tab1 SET At1 = At1*1.5 WHERE At2 LIKE 'B%';
1 row updated.
SQL> SELECT * FROM Tab1;
```

Натижа:

AT1	AT2
1	CCC
3	BBB

Листинг жадвал сатрларини такомиллаштириш доир мисол.

Махсус предикатлар (шарт параметрлари)

Маълумотларни танлаб олишни бажаришда WHERE калит сўз билан махсус предикатлар (параметр) IN, BETWEEN, LIKE, EXIST, IS NULL ишлатилади.

IN параметри: Предикатни ростлигини аниқловчи тўпламни аниқлайди.

Синтаксиси:

```
Предикат IN::= ифода [NOT] IN [қисмсўров|[қийматлар_ рўйхати]}
```

Ишлатганда эътибор бериш керак:

✓ *Чап операнд ифодани тоифаси ва ўнг томондаги операнд рўйхатидаги қийматлар тоифаси бўйича солиштириладиган бўлиши керак*

✓ *Қисм сўров натижасидаги ёзувлар тўплами фақат битта устунга эга бўлиши керак.*

✓ *IN параметрли сўров одатда ошкор ҳолда берилган кичик ўлчамли тўпламга ёки қисм сўров яратган тўпламга тегишли эканлигини текширишда ишлатилади.*

Мисол: Қуйидаги жадвал яратилган ва тўлдирилган бўлсин.

```
CREATE TABLE Tab1 (At1 CHAR(3), At2 NUMBER);
INSERT INTO Tab1 VALUES ('A1, 1);
INSERT INTO Tab1 VALUES ('B', 2);
INSERT INTO Tab1 VALUES ('C', 3);
```

IN параметри билан мисол:

```
SQL> SELECT * FROM Tab1 WHERE At1 IN ('A','C','E') AND At2 NOT IN
(2,4,8);
```

Натижа:

At1	At2
A	1
C	3

Листинг. Элементлпри ошкор санаб ўтилган тўпламли ва қисм сўров билан ҳосил қилинган тўпламли IN.предикатли сўровларни ташкил қилишга доир мисол.

```
SQL>SELECT * FROM Tab1 WHERE At1 IN (SELECT At1 FROM Tab1
WHERE At1 > 'A');
```

Натижа:

At1	At2
B	2
C	3

Between параметри

Between параметри қийматни берилган интервалга кириш ёки кирмаслигини аниқлайди. Синтаксиси:

BETWEEN::= ифода [**NOT**] **BETWEEN** бошланғич_қиймат **AND** охирги _қиймат

Мисол:

SQL> **SELECT** * **FROM** Tab1 **WHERE SIN**(At2) **BETWEEN** 1 **AND** 0;

LIKE параметри фақат CHAR, VARCHAR, VARCHAR2 тоифали майдонлар учун ишлатилади. Шартни яратиш учун шаблон(андоза) ишлатилади. У махсус ва оддий символлардан ташкил топади. Бунда:

- ✓ *«__» белги ихтиёрий битта символни алмаштиради.*
- ✓ *« % » белгиси ихтиёрий сондаги кетма кет символларни*

Синтаксиси:

LIKE::= атрибут_номи [**NOT**] **LIKE** шаблон

Мисол: Қуйидагича яратилган ва тўлдирилган жадвал берилган бўлсин

```
CREATE TABLE Tab1 (At1 VARCHAR2(3));
```

```
INSERT INTO Tab1 VALUES('AB');
INSERT INTO Tab1 VALUES('ABC');
INSERT INTO Tab1 VALUES('ACB1);
INSERT INTO Tab1 VALUES('ADC');
INSERT INTO Tab1 VALUES('CAB');
Мисол:
SQL> SELECT * FROM Tab1 WHERE At1 LIKE 'A%';
```

Натижа:

At1
AB
ABC
ACB
ADC

```
SQL> SELECT * FROM Tab1 WHERE At1 LIKE '%B';
```

Натижа:

At1
AB
ACB
CAB

```
SQL> SELECT * FROM Tab1 WHERE At1 LIKE 'A%B';
```

Натижа:

At1
ACB

```
SQL> SELECT * FROM Tab1 WHERE At1 LIKE 'A_C';
```

Натижа:

At1
ABC
ADC

IS NULL предикати

SQL тилида атрибутни аниқмас (номаълум) қийматини кўрсатиш Null ишлатилади. Атрибут қиймати Null бўлса, бу атрибут ҳеч қандай аниқ (конкрет) қиймат қабул қилмаганини англатади. Null қиймат махсус усулда кузатилади ва ҳеч қандай тоифага эга эмас. Ихтиёрий тоифали атрибут Null қийматга эга бўлиши мумкин.Аниқмас қийматларни қайта ишлаш учун SQL тилида махсус оператор IS Null ишлатилади. Синтаксиси:

```
Предикат IS NULL::= устун_номи IS [NOT] NULL IS [NOT]
```

NULL предикати хар доим TRUE ёки FALSE қийматни қабул қилади. Бунда фақат x аниқмас қийматга эга бўлса, x IS NULLифода TRUE қийматга тенг бўлади. x IS NOT NULL предикатини қиймати NOT(x IS NULL) қийматга тенг.

Мисол: Қуйидаги жадвал яратилган ва тўлдирилган бўлсин.

```
CREATE TABLE Tab1 (At1 CHAR(3), At2 NUMBER);
INSERT INTO Tab1 VALUES('A1', 1);
INSERT INTO Tab1 VALUES('B', NULL);
SQL> SELECT * FROM Tab1 WHERE At2 IS NULL;
```

Натижа:

At1	At2
B	

```
SQL> SELECT * FROM Tab1 WHERE At2 IS NOT NULL;
```

Натижа:

At1	At2
A	1

Қисм сўровлар

Қисм сўровлар яратишда EXITS предикати ишлатилади. Бу предикат рост қиймат қабул қилади агарда бирорта қисм сўров натижаси бўш бўлмаса. EXITS предикати автоном ҳолда ёки бошқа предикатлар комбинацияси билан мантиқий боғланишлар билан бирлаштириб ҳисобланиши мумкин. Бу предикат аниқмас қийматни кабул килиши мумкин эмас, яъни унинг қиймати ҳар доим TRUE ёки FALSE бўлади. Қиймат фақат TRUE бўлади, агарда қисм сўров ҳисоблаган натижа бўш тўплам бўлмаса.

Синтаксиси:

```
Предикат EXISTS::= EXISTS қисм сўров
```

Мисол: EXISTS ни ишлатишни қуйида яратилган ва тўлдирилган жадвалда намойиш қиламиз:

```
CREATE TABLE Tab1 (At1 CHAR(l), At2 NUMBER);
INSERT INTO Tab1 VALUES ('A',1);
INSERT INTO Tab1 VALUES ('B',2);
```

Бу мисолда EXISTS предикатини ишлатиб выполняется выборка всех строк таблицы Tab1, жадвални At2 атрибути кичик қиймат олган барча сатрларини танлаш бажарилади:

```
SQL> SELECT * FROM TaЫ a WHERE EXISTS (SELECT * FROM Tab1 b
WHERE a.At2 > b.At2);
```

Натижа:

AT1	***AT2***
B	*2*

Назарий саволлар:

1. Update қандай оператор?

2. Select қандай оператор?

3. Drop ва Delete ҳақида маълумот беринг.

4. Тасаввур нима?

5. Alter TABLE қандай вазифа бажаради?

5 – МАВЗУ

НАЗАРИЙ ТЎПЛАМ АМАЛЛАРИ.ТАШҚИ БИРЛАШТИРИШ. ТАРТИБЛАШ.

Режа:

1. *Назарий тўплам амаллари*
2. *Ташқи бирлаштириш*
3. *Тартиблаш*
4. *Гуруҳлаш ва агрегат функциялар*
5. *Шартсиз ва шартли ўртача қийматни ҳисоблаш*

Назарий тўплам амаллари

Назарий тўплам амаллари кўп компонентли сўров натижаларини битта натижага бирлаштиради. ORACLE бажарилиши мумкин бўлган амаллар қуйидагича талқин қилинади.

***UNION**- Берилган барча сўров натижаларини мунособат кўринишида (яъни такрорланувчи сатрларсиз) расмий бирлаштириш.*

***UNION ALL**-Берилган барча сўров натижаларини такрорланувчи сатрларни сақлаган ҳолда расмий бирлаштириш.*

***INTERSECT**- расмий кесишув; Сўровларни ташкил қилувчи барча натижаларга кирган сатрларни ўз ичига олади.*

***MINUS**- натижавий тўплам ўз ичига биринчи сўров натижасига кирган, лекин иккинчи сўров натижасига кирмаган сатрларни олади. Такрорланувчи сатрлар олиб ташланади. Барча назарий тўплам амаллари бир хил устунликка эга ва чапдан ўнгга қараб бажарилади.*

Назарий тўплам амалларида сўров натижаларини барча тўпламларида устунлар сони ва уларни тоифалари мос (келишилган) бўлиши керак.

```
CREATE TABLE Tab1 (At1 NUMBER);
CREATE TABLE Tab2 (At1 NUMBER);
INSERT INTO Tab1 VALUES (1);
INSERT INTO Tab1 VALUES (2);
INSERT INTO Tab1 VALUES (3);
INSERT INTO Tab2 VALUES (1);
INSERT INTO Tab2 VALUES (3);
INSERT INTO Tab2 VALUES (5);
```

Tab1

Alt1
1
2
3

Tab2

Alt2
1
3
5

Сўровларда назарий тўплам амалларини ишлатишга доир мисоллар:

```
SQL> SELECT * FROM Tab1 UNION SELECT * FROM Tab2;
```

At1
1
2
3
5

```
SQL> SELECT * FROM Tab1 UNION ALL SELECT * FROM Tab2;
```

At1
1

2
3
1
3
5

```
SQL> SELECT * FROM Tab1 INTERSECT SELECT * FROM Tab2;
```

At1
1
3

```
SQL> SELECT * FROM Tab1 MINUS SELECT * FROM Tab2;
```

At1
2

Ташқи бирлаштириш

Агар танлашни бажаришда WHERE калит сўз ишлатилмаса, унда натижа жадвал танлашда иштирок этаётган жадвалларни декарт кўпайтмасидан иборат бўлади .Бунда, одатда биринчи жадвални барча устунлари, иккинчи жадвални барча устунлари билан комбинациясини олиш талаб этилмайди, шунинг учун WHERE калит сўз билан аниқланадиган танлаш мезони ишлатилади. Бундай бирлаштириш оддий (simple join) деб аталади. Декарт кўпайтма ва оддий бирлаштиришни амалларини бажариш техникасини қуйидаги командалар ёрдамида яратилган ва тўлдирилган жадвалларда намойиш қиламиз.

Икки жадвалдан тузилган декарт кўпайтма ва оддий бирлаштириш амали сўровларга мисол

```
CREATE TABLE Tab1 (At1 NUMBER, At2 VARCHAR2(1));
CREATE TABLE Tab2 (At1 NUMBER, At2 VARCHAR2(1));
INSERT INTO Tab1 VALUES (1, 'A');
INSERT INTO Tab1 VALUES (2, 'B') ;
INSERT INTO Tab1 VALUES (3, 'C');
INSERT INTO Tab2 VALUES (1, 'a');
INSERT INTO Tab2 VALUES (3, 'b');
INSERT INTO Tab2 VALUES (5, 'c');
```

Tab1

Alt1	Alt2
1	A
2	B
3	C

Tab2

Alt1	Alt2
1	a
3	b
5	c

Tab1 ва Tab2 жадвалларни декарт кўпайтмаси амалига доир мисол

```
SQL> SELECT Tab1.At1,Tab1.At2,Tab2.At1,Tab2.At2 FROM TAB1,TAB2;
```

Alt1	Alt2	Alt1	Alt2
1	A	1	a
2	B	1	a
3	C	1	a
1	A	3	b
2	B	3	b
3	C	3	b
1	A	5	c
2	B	5	c

3	C	5	c

Tab1 ва Tab2 жадвалларни At1 атрибут қийматларини тенглиги мезони бўйича оддий бирлаштириш амалига доир мисол

```
SQL> SELECT * FROM Tab1,Tab2 WHERE Tab1.At1=Tab2.At1;
```

Alt1	**Alt2**	**Alt1**	**Alt2**
1	A	1	a
3	C	3	b

Ташқи бирлаштириш (outer join) оддий бирлаштиришга нисбатан кўпроқ сатрларни танлаб олади. Ташқи бирлаштириш амалини бажарганда, оддий бирлаштириш амали бажарилганда олинадиган барча устунлар олинади ва бир жадвалдан ишлатилаётган мезон бўйича бошқа жадвални бирорта ҳам сатрига мослик аниқланмаган қўшимча сатрлар ажратилади. Ташқи бирлаштириш WHERE иборани конструкциясида куйидаги икки шаклни бири кўриниши учрайди:

```
[жадвал1.]устун=[жадвал2.]устун(+)
[жадвал2.]устун(+)=[жадвал1.]устун
```

Ташқи бирлаштириш символи (+) ташқи бирлаштириш амали бажарилаётган устунга нисбатан, бевосита уни кетидан келади. Ҳар хил жадвал устунларида бериладиган ташқи бирлаштиришга доир мисол:

```
SQL> SELECT * FROM Tab1,Tab2 WHERE Tab1.At1 = Tab2.At1(+);
```

Alt1	**Alt2**	**Alt1**	**Alt2**

1	A	1	a
2	B		
3	C	3	b

```
SQL> SELECT * FROM Tab1,Tab2 WHERE Tab1.At1(+) = Tab2.At1;
```

Alt1	**Alt2**	**Alt1**	**Alt2**
1	A	1	a
3	C	3	b
		5	c

Тартиблаш

Сўров натижаларини ўсиб бориш ёки камайиб бориш тартибида (кетма кетликда) тартиблаш учун ORDER BY калит сўзи ишлатилади.Бу калит сўз ишлатилмаса, сатрлар ихтиёрий тартибда чиқарилади. Тартиблашни беришда қуйидаги синтаксисдан фойдаланамиз.

```
ORDER BY{ифода|ҳолат|устунни_альтернатив_номи}[ASC|DESC][,...]
```

Ифода параметри SELECT калит сўзидан кейин санаб ўтилган бир ёки бир неча устун асосида қурилган ифода қийматларини қабул қилади. Ҳолат параметри SELECT калит сўзидан кейин санаб ўтилган устун позицияларини идентификацияловчи сонни беради, яъни устун номларини кўрсатиш ўрнига SELECT рўйхатидаги устун позициясини номерини кўрсатиш мумкин. ASC ва DESC калит сўзлар мос равишда тартиблашни ўсиб бориш ёки камайиб бориш кетма кетликда(тартиблашни) жойлаштиришни кўрсатади.

Мисол: Tab1 жадвал берилган ва тўлдирилган

```
CREATE TABLE Tab1 (At1 NUMBER, At2 VARCHAR2(1));
INSERT INTO Tab1 VALUES (1, NULL);
```

```
INSERT INTO Tab1 VALUES (1, 'A’);
INSERT INTO Tab1 VALUES (2, 'B');
INSERT INTO Tab1 VALUES (3, 'C');
INSERT INTO Tab1 VALUES (3, 'A');
```

Tab1

At1	At2
1	Null
1	A
2	B
3	C
3	A

*Тартиблаш рўйхатида иккита At1*10 At2 ифода билан берилган тартиблаш мисолини кўриб чиқамиз:*

```
SQL> SELECT At1*10, At2 FROM Tab1 ORDER BY At1 ASC, At2 DESC;
```

Tab1

At1	At2
10	Null
10	A
20	B
30	C
30	A

Гуруҳлаш ва агрегат функциялар

Ажратиб олинган маълумотларни биргаликда қайта ишлов мақсадида гуруҳлаш ташкил қилинади ва бунинг учун GROUP BY калит сўзидан фойдаланамиз.Маълумотларга ишлов бериш одатда бирорта функция

ҳисоблашда (йиғинди, ўрта қиймат, ёзувлар сони ва ҳоказолар ишлатилади. Сатрларни гуруҳлаш конструкциясини синтаксиси:

GROUP BY ифода [, ифода] [HAVING шарт]

Ифода элементи - константа, атрибут ёки уларни функцияси бўлиши мумкин. GROUP BY ва HAVING калит сўзларни ишлатишни қуйидаги командалар ёрдамида яратилган жадвалда намойиш қиламиз:

```
CREATE TABLE Tab1 (At1 NUMBER, At2 NUMBER);
INSERT INTO Tab1 VALUES(1, 1);
INSERT INTO Tab1 VALUES(1, 2);
INSERT INTO Tab1 VALUES(2, 3);
INSERT INTO Tab1 VALUES(2, 4);
```

Tab1

At1	At2
1	1
1	2
2	3
2	4

Қуйидаги сўровлар гуруҳлашни қўшимча шартлар ва шартларсиз намойиш қилади.

SQL> **SELECT** At1, **AVG**(At2) “Ўртача At2" **FROM** Tab1 **GROUP BY** At1;

Tab1

At1	Ўртача At2
1	1.5
2	3.5

Шартсиз ва шартли ўртача қийматни ҳисоблаш

Гуруҳли функциялар SELECT операторини GROUP BY калит сўзи ишлатилиб сўров натижасида яратилган сатрлар гуруҳи бўйича ҳисобланган натижани қайтаради. Гуруҳли функциялар бўйича ҳисоблаш бажаришда сукут билан ALL калит сўзи ишлатилади. У такрорланувчи қийматларни ҳам олинишини англатади. Агар сўровда DISTINCT ишлатилса, унда гуруҳли функциялар атрибутларни ёки ифодаларни бир биридан фарқли қийматлари қаралади. Барча гуруҳли функциялар COUNT(*) дан ташқари атрибутларни ҳисоблашда аниқмас қийматларни (NULL) ҳисобга олмайди.Сонли атрибутларни ноаниқ қийматини 0 билан алмаштириш учун NVL қурилган стандарт функциядан фойдаланилади.

Гуруҳли функцияларни ишлатишни барчасини қуйидаги командалар билан яратилган ва тўлдирилган жадвалда намойиш қиламиз. HAVING калит сўзи билан берилган шарт, атрибутларни конкрет қийматига эмас, балки GROUP BY шарти билан яратилган гуруҳга тегишли бўлади. Агар танлаш шарти гуруҳга тегишли бўлмай, атрибутларга тегишли бўлса, унда у WHERE калит сўзидан кейин кўрсатилиши керак. Буни қуйидаги мисолда намойиш қилиш мумкин.

```
SQL>SELECT At1, AVG(At2) “Ўртача At2" FROM Tab1 GROUP BY At1
HAVING At2 > 1;
*
ERROR at line 2: ORA-00979: not a GROUP BY expression'
SQL>SELECT At1, AVG(At2) “Ўртача At2" FROM Tab1 WHERE At2 > 1
GROUP BY At1;
```

AT1	**Ўртача At2**
1	2

2	2.5

Гуруҳли функцияларни ишлатишни барчасини қуйидаги командалар билан яратилган ва тўлдирилган жадвалда намойиш қиламиз.

```
CREATE TABLE Tab1 (At1 NUMBER);
INSERT INTO Tab1 VALUES(1);
INSERT INTO Tab1 VALUES(1);
INSERT INTO Tab1 VALUES(2);
INSERT INTO Tab1 VALUES(NULL);
```

Tab1

At1
1
1
2
NULL

AVG ўрта қийматни ҳисоблаш функцияси

Ўрта қийматни ҳисоблаш функцияси AVG, NULL қийматни киритмаган ҳолда сонли аргументли ифодани ўрта қийматини ҳисоблаб қайтаради.

Команда синтаксиси:

```
AVG([DISTINCT|ALL] ифода)
```

Мисоллар:

```
SQL>SELECT AVG(At1) "(1+1+2)/3" FROM Tab1;
```

(1+1+2)/3
1.333333

```
SQL>SELECT AVG(DISTINCT At1) "(1+2)/2" FROM Tab1;
```

(1+2)/2
1.5

```
SQL>SELECT AVG(NVL(At1,0)) "(1+1+2+0)/4" FROM Tab1;
```

(1+1+2+0)/4
1

Йиғинди, дисперсия ва дисперсия квадратини ҳисоблаш. Йиғинди SUM функцияни ҳисоблаш синтаксиси:

```
SUM({DISTINCT|ALL] ифода)
```

Дисперсия STDDEV ҳисоблаш функция синтаксиси:

```
STDDEV([DISTINCT|ALL] ифода)
```

Сонли атрибут қийматларини дисперсияси квадратини ҳисоблаш. VARIANCE функция синтаксиси:

```
VARIANCE([DISTINCT|ALL] ифода)
```

Мисоллар:

```
SQL>SELECT SUM(At1),STDDEV(At1),SQRT(VARIANCE(At1)) FROM
Tab1;
```

SUM(At1)	STDDEV(At1)	SQRT(VARIANCE(At1))
4	.57735027	.57735027

COUNT функцияси.

Танлаб олинган сатрларни сонини чиқариш учун COUNT функцияси ишлатилади. Алоҳида кўриниши COUNT(*) жадвалдаги дубликатлар ва аниқмас NULL қийматли сатрларни қўшган ҳолда чиқаради. Оператор синтаксиси:

```
COUNT([DISTINCT|ALL] ифода)
```

Мисоллар:

```
SQL>SELECT COUNT(DISTINCT At1),COUNT(At1),COUNT(*) FROM
Tab1;
```

COUNT(DISTINCT AT1)	COUNT(AT1)	COUNT(*)
2	3	4

Энг катта (энг кичик) MAX (MIN)) қийматни танлаш функциялари:

Энг катта MAX функция синтаксиси:

```
MAX([DISTINCT|ALL] ифода)
```

Энг кичик MIN функция синтаксиси:

```
MIN([DISTINCT|ALL] ифода)
```

Мисолларни қуйидаги командалар билан яратилган ва тўлдирилган жадвалларда кўриб чиқамиз.

```
CREATE TABLE Tab1(At1 VARCHAR2(1),At2 DATE);
INSERT INTO Tab1 VALUES('A','15-06-2001');
INSERT INTO Tab1 VALUES('B','21-09-2001');
```

```
INSERT INTO Tab1 VALUES('C',NULL);
```

At1	At2
A	15-06-2001
B	21-09-2001
C	NULL

Мисол: Устунни қийматларидан максимал ва минимал аниқлаш мисоли листинги.

```
SQL>SELECT MAX(At1),MIN(At2) FROM Tab1;
```

MAX(At1)	MIN(At2)
C	15-10-2011

Назарий саволлар:

1. *Жадвалдаги ёзувлар сонини аниқлаш сўровини ёзиб беринг.*
2. *Жадвалнинг бирон устунидаги энг катта элементни топиш сўровини айтиб беринг.*
3. *Жадвалнинг бирон устунидаги энг кичик элементни топиш сўровини айтиб беринг.*
4. *Жадвалдаги хар ҳил қийматлардан иборат бўлган элементлар йиғиндисини топинг.*

6 – МАВЗУ

ORACLE ДА УЗОҚЛАШГАН МБ АЛОҚА ЯРАТИШ (ЎРНАТИШ). КЕТМА – КЕТЛИКЛАР. СИНОНИМЛАР ВА УЛАРНИ ЯРАТИШ.

Режа:

1. *Узоқлашган Oracle маълумот базаси билан боғланиш яратиш*
2. *Кетма – кетлик*
3. *ORACLE да синонимлар яратиш*

Узоқлашган Oracle маълумот базаси билан боғланиш яратиш

Узоқлашгвн мвълумот базаси билан боғланиш ўрнатиш учун SQL – оператори CREATE DATABASE LINK командаси ишлатилади. Бунда локал ва узоқлашган маълумот базасида махсус дастурий таъминот ўрнатилган бўлиши керак. Узоқлашган маълумот базаси билан боғланиш операторини синтаксиси қуйидагича:

CREATE [**PUBLIC**] **DATABASE LINK** МБбиланбоғланиш_номи [**CONNECT TO** фойдаланувчи_номи **IDENTIFIED BY** фойдаланувчи_пароли] **USING** ‘боғланиш _сатри'

Мисол: sun_ora_link боғланиши аниқланан мисолни кўриб чиқамиз.БДу боғланиш u1 фойдаланувчи ҳисобот ёзувини u1psw паролини ишлатади. sunora боғланиш сатри билан аниқланаган.Узоқлашган маълумотлар базаси билан муваффақиятли боғланиш ўрнатилгандан кейин, уни жадвалларига сўровлар бажариш мумкин.

SQL> **CREATE DATABASE LINK** sun_ora_link **CONNECT TO** ul **IDENTIFIED BY** ulpsw **USING** 'sunora';
Database link created.
SQL> **SELECT * FROM** Tab1@sun_ora_link;

At1
1
2

Фойдаланувчидан u1 фойдаланувчини Tab1 жадвали жойлашганини беркитиш (скрыть) учун синонимдан фойдаланиш мумкин. Мисол: Қуйидаги мисолда узоқлашган маълумот базаси билан ошкормас ҳолда боғланишни таъминлаш учун ва ундан фойдаланиш сўровини листинги келтирилган.

```
SQL> CREATE SYNONYM suntab1 FOR ul.Tab1@sun_ora_link;
 Synonym created.
SQL> SELECT * FROM suntab1;
```

At1
1
2

DROP DATABASE LINK командаси. Бу команда узоқлашаган маълумот базаси билан боғланишни узуш учун ишлатилади. Бу команда синтаксиси:

```
DROP [PUBLIC] DATABASE LINK МБ боғланиш_номи
```

Мисол: Sun_ora_link номли узоқлашган маълумотлар базаси билан боғланиш олиб ташлаш.

```
SQL> DROP DATABASE link sun_ora_link;
Database link dropped.
```

Кетма – кетлик

Такрорланмайдиган бутун сонларни генерация қиладиган маълумотлар базаси объекти кетма кетлик дейилади.Кетма – кетлик натижасида ҳосил қилинган сонлар одатда бирламчи калит қиймати бўлиб ишлатилади. Кетма

кетдлик натижасида ҳосил қилинган сонлар доим ўсиши, ёки маълум чегаргача ўсиши ёки чегарага етгандан бошлаб яна кайтадан ўсиши мумкин.Шунингдек орттирмани қийматини ҳам бериш мумкин.

Кетма кетлик аниқлаш оператори синтаксиси:

CREATE SEQUENCE [схема_номи.] кетмакетлик_номи [**INCREMENT BY** орттирма] [**START WITH** бошланғич_қиймат] [**MAXVALUE** энг_катта_қиймат|**NOMAXVALUE**][**MINVALUE** энг_кичик_қиймат|**NOMINVALUE**] [**CYCLE**|**NOCYCLE**][**CACHE** элементлар_сони|**NOCACHE**][**ORDER**|**NOORDER**]

INCREMENT BY параметри кетма кет номер орасидаги интервал беради. START WITH калит сўз бошланғич қиймат параметри орқали биринчи генерация қилинган номерни беради. Агар бу параметр берилмаса, ўсувчи кетма кетлик учун бошланғич генерация қиладиган номер MINVALUE қийматига (камаювчи учун MAXVALUE) тенг қилиб олинади.

Seq1 кетма – кетликни яратишни кўриб чиқамиз. Кетма – кетликни бошланғич элементи 2 га тенг деб аниқланаган, энг_катта_қиймат энг_кичик_қиймат параметри мос равишда 3 ва 1 тенг деб аниқланган.

```
SQL> CREATE SEQUENCE Seq1 MAXVALUE 3 MINVALUE 1 START
WITH 2;
Sequence created.
SQL> SELECT Seq1.NEXTVAL FROM dual;
```

NEXTVAL
2

```
SQL> SELECT Seq1.NEXTVAL FROM dual;
```

NEXTVAL
3

```
SQL> SELECT Seq1.NEXTVAL FROM dual;
ERROR:
ORA-08004: sequence SEQl.NEXTVAL exceeds MAXVALUE
and cannot be instantiated no rows selected
```

Олдинги мисолни ўзгартириб Seq2 кетма кетлигини яратамиз. Кетма кетликни бошланғич элементи, энг_катта_қиймат, энг_кичик_қиймат параметри ҳам олдинги мисолдаги каби, фақат қўшимча қилиб цикл параметри 2 га тенг, ва кэшлаш параметри 2 га тенг деб аниқланган. (агар шундай қилиб кэшлаш параметри аниқланмаса, хатолик пайдо бўлади, чунки CACHE калит сўзини элементлар_сони параметри MAXVALUE ва MINVALUEкалит сўзлар билан бериладиган параметрларини айирмасидан катта бўлмаслиги керак).

```
SQL> CREATE SEQUENCE Seq2 MAXVALUE 3 MINVALUE 1 START
WITH 2 CYCLE CACHE 2;
Sequence created.
SQL> SELECT Seq2.NEXTVAL FROM dual;
```

NEXTVAL
2

```
SQL> SELECT Seq2.NEXTVAL FROM dual;
```

NEXTVAL

3

```
SQL> SELECT Seq2.NEXTVAL FROM dual;
```

NEXTVAL
1

DROP SEQUENCE командаси кетма кетликни олиб ташлаш учун ишлатилади. Команда синтаксиси:

```
DROP SEQUENCE [схема_номи.]кетма_кетлик_номи
```

Мисол: Кетма кетликни олиб ташлаш мисол

```
SQL> DROP SEQUENCE Seq1;
Sequence dropped.
```

ORACLE да синонимлар яратиш

Синоним алтернатив номлаш учун ишлатиладиган маълумот база объекти ҳисобланади. Одатда синоним жадваллар, тасаввурлар, умумий фойдаланишга мўлжалланган (схема префиксни кўрсатмаган ҳолда) кетма кетликларни ёки узоқлашган маълумотлар базасига мурожатни беркитиш учун яратилади. Синтаксиси:

```
CREATE [PUBLIC] SYNONYM [схема_номи.]синоним_номи FOR
[схема_номи2.]объект_номи [@имя_ связиБД]
```

Мисол: RefCodes жадвали фойдаланувчи Administrator томонидан берилган ва қуйидаги командалар билан тўлдирилган бўлсин.

```
CREATE TABLE RefCodes(At1 NUMBER, At2 VARCHAR2(5));
INSERT INTO RefCodes VALUES(1,'Textl');
INSERT INTO RefCodes VALUES(2,'Text2');
```

SYSTEM фойдаланувчи u1 фойдаланувчига, фойдаланувчи Administrator схемасидан RefCodes жадвали мурожат қилган ҳолда синоним яратади. Бунда синомимни тўғрилигига қарамасдан, синоним мурожат қилаётган объект, фойдаланувчи u1 тақиқланган (недоступен).

Синоним яратишга доир мисол

```
SQL> CONNECT system/manager@sunora;
Connected.
SQL> CREATE SYNONYM ul.RefCodes FOR Administrator.RefCodes;
Synonym created.
SQL> CONNECT ul/ulpsw@sunora;
Connected.
SQL> SELECT * FROM RefCodes;
SELECT * FROM REFCODES * ERROR at -line 1: ORA-00942: TABLE or
view does not exist
```

Назорат саволлари

1. *Кетма – кетликлар қандай яратилади?*
2. *Синонимлар қандай яратилади?*
3. *Узоқлашган маълумотлар базасидаги жадвалнинг устида қандай қилиб амаллар бажарилади.*

7 – МАВЗУ

PL/SQL – SQL ТИЛИНИНГ ПРОЦЕДУРАЛИ КЕНГАЙТМАСИ. PL/SQL ДА ДАСТУР ТУЗИЛИШИ (СТРУКТУРАСИ). ЎЗГАРУВЧИЛАР, КОНСТАНТАЛАР ВА ТОИФАЛАР.

Режа:

1. *PL/SQL да дастур тузилиши (структураси)*
2. *Блоклар ҳақида*
3. *Ўзгарувчилар константалар ва тоифалар*
4. *Ўзгарувчига қиймат бериш*
5. *RECORD мураккаб тоифаси*

PL/SQL да дастур тузилиши (структураси)

PL/SQL да дастури одатда 3 та блокдан ташкил топади: тавсифлаш блоки, бажариш блоки ва фавқулода вазиятларни қайта ишлаш блоки. Бажариш блоки BEGIN ва END оператор қавсларини ишлатиш билан структуралаштирили мумкин. PL/SQL да дастур синтаксис қуйидагича расмийлаштирилиши мумкин

```
DECLARE
    операторлар…
BEGIN
    операторлар…
EXCEPTION
    операторлар…
END;
```

Бундай кўринишдаги блок аноним блок дейилади.

Блоклар ҳақида

DECLARE блокида ўзгарувчилар константалар ва фойдаланувчилар томонидан аниқланадигна маълумотлар тавсифланади (эълон қилинади) тавсифланадиган. Биринчи BEGIN оператори асосий дастурни жисмини бошланишини белгилайди. Дастур жисмига бошқа BEGIN ва END операторлар қавслари билан чегараланган бошқа блоклар киритилиши мумкин. EXCEPTION блокида дастурда фавқулода вазиятларни кайта ишлов учун дастур кодини фрагменти берилади. Охирги END оператори дастур жисмини охирини кўрсатади. Аноним блокда DECLARE ва EXCEPTION блоклари бўлмаслиги мумкин, лекин BEGIN ва END оператор қавслари билан чегараланган операторлар блоки бўлиши керак. Баъзи ностандарт ҳолларларда (вырожденном) фақат NULL келиши мумкин.

PL/SQL дастурини ихтиёрий қисмида изоҳ (шарҳ) келтириш мумкин. «--» символ билан бошланадиган ва жорий сатр охиригача давом этадиган ихтиёрий матн изоҳ ҳисобланади. Кўп сатрли изоҳлар « / » ва «*/ » символлар орасига олиб ёзилади. Дастурда изоҳлар келтириш дастурлаш амалиётини яхши услубияти ҳисобланади. Шунингдек дастурларни ёзишда уларни ўқишни енгиллаштириш мақсадида сатр бошига нисбатан силжитиб ёзиш ҳам, яхши услубият ҳисобланади.*

DECLARE блоки олдида ҳар хил дастурий воситалар учун ўзгарувчилар муҳитини ўрнатиш командалари келиши мумкин.

Ўзгарувчилар константалар ва тоифалар

PL/SQL да дастури DECLARE ва BEGIN операторлари билан чекланган блокида ўзгарувчилар, тоифалар(типлар) ва константалар тавсифланади. Ихтиёрий ўзгарувчи ёки константа PL/SQL да мумкин бўлган тоифалардан бирига эга бўлади.. PL/SQL да маълумот тоифалари юқорида кўриб чиқилган SQL даги маълумот тоифалари (VARCHAR2, NUMBER, DATE, BOOLEAN ва бошқалар) билан (баъзи бир муҳим бўлмаган фарқларни эътиборга олмаса, маълумотларни максимал узунлигидаги фарқ, PL/SQL учун махсус маълумот тоифалари PLS_INTEGER, BINARY_INTEGER ва бошқалар) бир хил деб

ҳисобласа бўлади. Шу билан бирга PL/SQLда мураккаб маълумот тоифалари (RECORD, массивлар ва PL/SQL –жадваллари) ишлатилади. Константа CONSTANT калит сўз билан идентификацияланади ва ўзгарувчидан уни қийматини ўзгартиришга ҳаракат қилинганда, хатолик чиқариш билан фарқ қилади. Ўзгарувчига қиймат бериш «:=» оператор билан амалга оширилади.

Ўзгарувчига қиймат бериш

Ўзгарувчиларга қиймат бериш барча дастур тилларида муҳим ўрин эгаллайди. ORACLE да куйидаги конструкциялар мавжуд:

```
Ўзгарувчи_номи ўзгарувчи_тоифаси:=қиймат;
Ўзгарувчи_номи:=қиймат;
```

Мисол: Дастурни эълон килиш бўлими DECLARE

```
Counter number:=0;
```

Ўзгарувчини инициализация қилинаяпти

```
Today date:=sysdate;
```

Ўзгарувчига тизим санаси берилаяпти

```
Name varchar2(25);
```

Ўзгарувчи эълон қилинаяпти, лекин қиймат берилмаяпти

RECORD мураккаб тоифаси

RECORD мураккаб маълумот тоифаси ёзувларни сақлаш ва қайта ишлаш учун мўлжалланган. Ҳар бир ёзув атрибутларга эга бўлиб, уларга бошланғич қиймат бериш (инициализация қилиш) ёзувни эълон қилишда амалга оширилиши мумкин. Қиймат беришда ўзгарувчи номи ва атрибут номи нуқта билан ажратилиб биргаликда берилиши керак.

RECORD ўзгарувчи тоифасини курсорлар ёрдамида маълумотларни танлаб олишда ишлатиш қулай.

Назорат саволлари:

1. *Ўзгарувчига қиймат қандай ўзлаштирилади?*
2. *Структуралар қандай эълон қилинади?*
3. *Қиймат бериш операторини айтиб беринг?*

8 – МАВЗУ

ДАСТУРНИ БАЖАРИШНИ БОШҚАРИШ. ТАРМОҚЛАНИШ, ЦИКЛ, GOTO ОПЕРАТОРЛАРИ. МАХСУС ВАЗИЯТЛАРНИ ҚАЙТА ИШЛАШ. (5.PDF)

Режа

1. *Тил алфавити ва муҳит командалари*
2. *Шартли IF оператори*
3. *Case оператори*
4. *Цикллар*

Тил алфавити ва муҳит командалари

- ✓ *Барча катта ва кичик лотин ҳарфлари;*
- ✓ *Рақамлар: 0..9;*
- ✓ *Математик операторлар:*
  - ❖ *** - даражага кўтариш;*
  - ❖ **, / -кўпайтириш ва бўлиш;*
  - ❖ *+, - , ||- қўшиш, айириш, улаш амаллари;*

SET SERVEROUTPUT ва SET ECHO - фойдаланувчи терминалига чиқариш режимини аниқлаш командалари.

Процедура DBMS_OUTPUT.PUT_LINE процедураси маълумотларни фойдаланувчи терминалига чиқаришн таминлаб беради. Белги (символ) "/" - аноним блок матнини якунланганини ва командаларни таҳлил қилиш ва бажариш кераклигини, белгилайди.

Мисол: Ўзгарувчиларни аниқлаш ва 2 ва 3 сонларни логарифмини ҳисоблаш дастурини тузамиз.

```
SQL> SET SERVEROUTPUT ON;
SQL> SET ECHO ON;
```

```
SQL> DECLARE
        Headerl CONSTANT VARCHAR2(20) := ‘Ikkini logorifmi teng’;
        Header2 CONSTANT VARCHAR2(20) := ‘Uchni logorifmi teng’;
        Arg NUMBER := 2; -- Bu yrda argumentni qiymAT1 beriladi
        -- Bajariluvchi blok
        BEGIN
              DBMS_OUTPUT.PUT_LINE(Header1||LN(Arg));
              Arg :== Arg+1;
              DBMS_OUTPUT.PUT_LINE(Header2||LN(Arg));
        END;
Натижа:
Ikkini logorifmi teng
0.693147180559945309417232121458176568081 4
Uchni logorifmi teng
1.098612288668109691395245236922525704 66
PL/SQL procedure successfully completed.
```

Шартли IF оператори

Умумий холда IF операторини формати қуйидагича:

```
IF
        шарт1
THEN
        операторлар1; -- биринчи тармоқ
[ELSIF шарт2 THEN операторлар2;] -- иккинчи тармоқ
.....
[ELSE операторларN;] -- альтернатив тармоқ
END IF;
```

IF-THEN конструкцияси

```
IF
```

```
        шарт1
THEN
        операторлар;
END IF;
```

Мисол:

```
IF
        l_date >’10-sep-10’
THEN
        l_salary := l_salary *1.15; -- маош 15 % оширилади.
END IF;
```

Дастурда IF - THEN операторларини бир-бирига жойлаштириш мумкин.

Масалан:

```
IF
        l_date >’10-sep-10’
THEN
        IF name = ‘Ivanov’
        THEN
                l_salary := l_salary *1.15; --маош 15 % оширилади.
        END IF;
END IF;
```

IF-THEN-ELSE конструкцияси

```
IF
        шарт1
THEN
```

```
        операторлар;
ELSE
        операторлар;
END IF;
```

Масалан:

```
IF
        l_date > '10-sep-10'
THEN
        l_salary:=l_salary*1.15; -- маош 15% ошади
ELSE
        l_salary:=l_salary*1.05; -- маош 5% ошади
END IF;
IF
        name ='Ivanov'
THEN
        l_salary:=l_salary *1.15; --маош 15% ошади
ELSIF
        name='Aliev'
THEN
        l_salary:=l_salary*1.10; --маош 10% ошади
ELSE
        l_salary:=l_salary*1.05; -- маош 5% ошади
END IF;
```

Case оператори

Oracle 9i бошланиб case оператори PL/SQL таркибига киритилди. Унинг формати қуйидагича:

```
CASE ўзгарувчи
        WHEN ифода_1 THEN қиймат_1;
        WHEN ифода_2 THEN қиймат_2;
        WHEN ифода_3 THEN қиймат_3;
        WHEN ифода_4 THEN қиймат_4;
        .................
        WHEN ифода_n THEN қиймат_n;
        ELSE қиймат_(n+1);
END CASE;
```

Масалан,

```
DECLARE v NUMBER :=1;
BEGIN
        CASE v
                WHEN 1 THEN DBMS_OUTPUT.PUT_LINE('Бир');
                WHEN 2 THEN DBMS_OUTPUT.PUT_LINE('Икки');
                WHEN 3 THEN DBMS_OUTPUT.PUT_LINE('Уч');
                WHEN 4 THEN DBMS_OUTPUT.PUT_LINE('Тўрт');
                WHEN 5 THEN DBMS_OUTPUT.PUT_LINE('Беш');
        END CASE;
END;
```

Цикллар

PL/SQL тилида 3 хил тоифали цикллар ишлатилади;

- ✓ *LOOP цикл оператори;*
- ✓ *While цикл оператори;*

- ✓ *For сонли цикл оператори;*

LOOP цикл оператори

LOOP цикл оператори формати:

```
LOOP
    операторлар;
    EXIT [when шарт];
END LOOP;
```

Масалан:

```
DECLARE v_counter DINARY_INTEGER:=1;
BEGIN
    LOOP -- temp_Tab1t jadvaliga schotchik
        -- qiymatlarini kiritamiz
        INSERT INTO temp_TABLE VALUES(v_counter,‘loop index’);
        v_counter:=v_counter+1;
        EXIT WHEN v_counter >50;
    END LOOP;
END;
```

While цикл оператори.

Бу операторни формати қуйидагича:

```
WHILE шарт LOOP
    оператор1;
    оператор2;
    . . . . . .
    операторN;
END LOOP;
```

Масалан:

```
DECLARE v_counter BINARY_INTEGER:=1;
```

```
    WHILE v_counter <=50 LOOP
        INSERT INTO temp_TABLE VALUES(v_counter, 'loop index');
        v_counter:=v_counter+1;
    END LOOP;
END;
```

For цикл оператори

Бу циклда қайтарилишлар сони маълум. Циклни формати куйидагича:

```
FOR «цикл_счетчиги» IN[REVERSE]«қуйи_чегара »..«юқори_чегара»
LOOP
    операторлар_кетма_кетлиги;
END LOOP;
```

Бу ерда: цикл счетчиги – ошкормас холда яратиладиган индексли ўзгарувчи; қуйи чегара ва юқори чегара итерациялар (такрорланишлар) сонини кўрсатади; Операторлар кетма – кетлиги цикл мазмунини ташкил қилади; цикл чегаралари бар марта ҳисобланиб қайтаришлар сонини аниқлайди. Счетчик қуйи чегарадан юқори чегарагача 1 қўшиб ўзгаради. (цикл тугагунга қадар); счетчик(индекс) ёки цикл параметри ошкормас холда BINARI_INTEGER тоифали деб эълон килинади. Циклдан олдин уни эълон қилиш шарт эмас.

Мисол: жадвалини 1..50 сонлар билан тўлдириш.

```
BEGIN
    FOR v_counter IN 1..50 LOOP
        INSERT INTO temh_tadle VALUES(v_counter,'loop_index');
    END LOOP;
END;
```

FOR циклида REVERSE калит сўзи ишлатилса цикл индеси (тескари тартибда) юкори чегарадан куйи чегарагача узгаради. Бунда чегараларни жойлашиши узгаришсиз колади, яъни пастки чегара биринчи кўрсатилади.

```
BEGIN
    FOR v_counter IN REVERSE 10..50 LOOP
        //v_ counter 50 дан бошланиди ва 1 га камайиб боради
        NULL;
    END LOOP;
END;
```

Цикл диапазонлари (юқори ва қуйи чегаралар) сонли ифодалар билан берилиши мумкин. Масалан:

```
DECLARE
    v_lowvalue NUMBER:=10;
    v_highvalue NUMBER:=40;
BEGIN
    FOR v_counter IN REVERSE v_lowvalue..v_highvalue LOOP
        INSERT INTO temp_Tab1e VALUES(v_counter,‘dinavic’);
    END LOOP;
END;
```

Назорат саволлари

1. *Тил алфавити ва муҳит командалари санаб беринг.*
2. *Шартли IF оператори ишлаш жараёнига мисол келтиринг.*
3. *Case оператори ишлаб жараёнини изоҳланг.*
4. *Ҳар бир цикл операторларига ишлаб жараёнига мисоллар келтиринг.*

9 – МАВЗУ

ПРОЦЕДУРАЛАР, ФУНКЦИЯЛАР ВА ПАКЕТЛАР. ORACLE НИ SQL ФУНКЦИЯЛАРИ. СИМВОЛЛАР ВА СОНЛИ КОДЛАР ОРАСИДА МОСЛИК ЎРНАТУВЧИ ФУНКЦИЯЛАР.

Режа:

1. *Процедура, функция ва пакетлар.*
2. *Умумий тушунчалар*
3. *Фойдаланувчи процедура ва функцияларини яратиш*
4. *Oracle процедурасини аниқлаш операторини синтаксиси*
5. *Процедурани бажариш*

Процедура, функция ва пакетлар

PL/SQL тилида процедура, Функция ва пакетлар кенг ишлатилади. Процедура (procedure) аниқ амалларни бажаришга мўлжалланган биргаликда ишлатиладиган SQL ва PL/SQL, тилларининг операторлари ,ўзгарувчилар ва тоифалар тўпламидан иборат дастур.

Функциялар(functions) аниқ амалларни бажаришга мўлжалланган биргаликда ишлатиладиган SQL ва PL/SQL, тилларининг операторлари ,ўзгарувчилар ва тоифалар тўпламидан иборат дастур.

Процедура ва функцияни фарқи катта эмас . Функция хар доим чақирувчи дастурга қиймат қайтаради. Процедура бундай қилмайди.

Пакетлар (packages) бу процедура, функция, ўзгарувчи ва SQL операторлар коллекциясидан иборат бўлиб , биргаликда гуруҳланган ва ягона дастур модули кўринишида сақланади. Процедура ва функциялар маълумот база объектлари бўлиб келади.

Фойдаланувчи процедура ва функцияларини яратиш

Процедура ва функциялар – маълумот база объектларидир ва , демак улар Create командаси билан яратилади ва Drop командаси билан йўқотилади. Процедура ёки функцияларни яратишда маълумот объекти номлари, параметрларни тоифаларива рўйхати ва PL/SQL тилида кодланган дастурни ишлаш мантиқи аниқланган бўлиши керак. Процедура ва функциялар яратиш учун Create procedure тизим имтиёзларига эга бўлиш керак. Янги процедура ва функция номи аниқлангандан сўнг, унинг номини. тоифасини ва параметрларини кўринишини бериш керак. Хар бир параметр учун одатда уни кўриниши IN, OUT, IN OUT калит сўзлар билан кўрсатилади.

IN кўринишли параметрда, параметрни қиймати дастурга мурожат қилганда аниқланади ва дастурда ўзгартирилмайди.

OUT параметр кўриниши, дастурни ишлаш жараенида параметр қийматини ўзгартириш мумкинлигини англатади.

IN OUT параметрига чақиришда қиймат тайинланади ва у дастур танасида (жисмида) ўзгартирилиши мумкин. Функция таърифида, процедура таърифига қўшимча, функция кайтараетган (чиқараётган) қийматни тоифаси кўрсатилади. Функция қийматини қайтаришни RETURN оператори бажаради.

Oracle процедурасини аниқлаш операторини синтаксиси

Create [or replace] procedure процедура_номи [(параметр_номи[{IN | OUT| |IN OUT }] маълумот_тоифаси [(параметр_номи [{IN | OUT J IN OUT}] маълумот_тоифаси ...])'] {IS I AS} PL/SQL_да_дастур;

OR REPLACE калит сўзи процедурани эски матнини шартсиз алмаштиришни кўрсатади. Агар OR REPLACE калит сўзи кўрсатилмаса ва процедура аниқланса, процедура кодини эски қийматини алмаштириш бўлмайди ва хатолик ҳақида ахборот чиқарилади. Маълумотларни аниқлаш

блоки AS (ёки IS фойдаланувчини танлашига қараб) калит сўзидан кейин бошланади.

Мисол: Жадвалга сонли параметр ва жорий санага боғлиқ функция қийматларини киритиш процедурасини ятинг.

Масалани ечиш учун Tab1 жадвали яратилган бўлсин:

```
Create Table Tab1 (At1 number, At2 date);
```

Дастур листинги:

```
SQL> CREATE OR REPLACE PROCEDURE InsRec(Arg1 in number) AS
Coeff constant number:=0.5;
BEGIN
        INSERT INTO Tab1 VALUES(coeff*Arg1, sysdate)
END;
procedure created
```

Процедурани бажариш

Процедурани бажарилиши натижасида маълумот базасидаги ўзгаришларини кўришни жадвални танлашни бажариб кўриш мумкин.

```
SQL> BEGIN
                InsRec(240)
        END;
/PL/SQL procedure successfully completed.
SQL> SELECT * FROM Tab1;
```

Назарий саволлар:

1. *Процедура яратиш буйруғини тушунтириб беринг.*
2. *Мавжуд процедурани ўзгартириш буйруғини тушунтириб беринг.*
3. *Процедуралар қандай вазифа бажаради.*

10 – МАВЗУ

СОНЛИ ФУНКЦИЯЛАР. САНАЛАР БИЛАН АМАЛ БАЖАРУВЧИ ФУНКЦИЯЛАР. ORACLE ФУНКЦИЯНИ АНИҚЛАШ ОПЕРАТОРИНИ СИНТАКСИСИ

1. *Функциядан фойдаланиш*
2. *Пакетлар*
3. *Oracle пакетини бажарилувчи қисмини (танасини) аниқлаш операторини синтаксиси*
4. *Пакетдан фойдаланиш*

Oracle функцияни аниқлаш операторини синтаксиси

CREATE [**OR REPLACE**] **FUNCTION** функция_номи
[(параметр_номи[{**IN**|**OUT**|**INOUT**}] маълумот_тоифаси [,параметр_номи
[{**IN**|**OUT**|**INOUT**}] маълумот_ тоифаси...])] **RETURN** маълумот_тоифаси
{**IS**|**AS**} PL/SQL_да_дастур;

Мисол: Функция параметрида берилган интервалга тушувчи саналарга мос келувчи атрибут кийматларини йиғиндисини ҳисоблаш функцияси яратинг.

Ечиш: Tab1 жадвали яратилган ва тўлдирилган бўлсин.

CREATE TABLE Tab1 (Atl **NUMBER**, At2 **DATE**);
INSERT INTO Tab1 **VALUES**(5, **SYSDATE**);
INSERT INTO Tab1 **VALUES**(6, **SYSDATE**);
INSERT INTO Tab1 **VALUES**(7, **SYSDATE**+1);

Дастур листинги:

```
SQL> CREATE OR REPLACE FUNCTION SumRecInt
     (Argl IN DATE, Arg2 IN DATE) RETURN NUMBER AS
     SumVar NUMBER := 0;
     BEGIN
```

```
        SELECT Sum(Atl) INTO SumVar FROM Tab1
        WHERE At2 BETWEEN Argl AND Arg2;
        RETURN SumVar;
    END;
Function created.
```

Функциядан фойдаланиш

```
SQL>BEGIN
    DBMS_OUTPUT.PUT_LINE(SumRecInt(SYSDATE-1/2,
SYSDATE+1/2));
    END;
PL/SQL procedure successfully completed.
```

Процедура ва функция ўчириб ташлаш учун мос равишда қуйидаги командалардан фойдаланамиз.

```
DROP PROCEDURE «процедура_номи»;
DROP FUNCTION «функция_номи»;
```

Пакетлар

Пакет объект сифатида олиб қаралганда икки қисмдан ташкил топади.

- *Пакет спецификацияси (интерфейси);*
- *Пакет танаси (жисми).*

Пакет спецификациясида процедура ва функциялар, глобал ўзгарувчилар, константалар, тоифалар, ташқи иловалар кириши учун курсор тавсифлар сақланади.

Пакет танасида барча процедуралар,функциялар ва ўзгарувчилар аниқланади. Пакет танасида аниқланган, лекин унинг спецификациясида тавсифланмаган процедура, функциялар, ўзгарувчилар локал ҳисобланади.

```
CREATE [OR REPLACE] PACKAGE пакета_номи (IS|AS)
PL/SQL_да_пакета_спецификацияси;
```

Мисол: Константалар, функциялар ва процедуралар тавсифидан ташкил топган пакет спецификациясини яратинг.

```
SQL> CREATE OR REPLACE PACKAGE PACAA AS
PACAA_CONST CONSTANT NUMBER := 1.2;
FUNCTION MULCONST(Argl NUMBER) RETURN NUMBER;
PROCEDURE AUDITMUL;
END;
Package created.
```

Oracle пакетини бажарилувчи қисмини (танасини) аниқлаш операторини синтаксиси

```
CREATE [OR REPLACE] PACKAGE BODY
Пакета_номи {IS|AS} PL/SQL_да_пакет_спецификацияси
```

Юқорида келтирилган листинг учун пакет танасини яратамиз. Фараз қиламиз, mulconst пакет функцияси аргументни пакет константасига купайтиришни бажарсин.

AUDITMUL процедура эса mulconst функциясига мурожатлар фактини фиксирласин. Mulconst жадвалини ёзувларини жадвалдаги қийматларни мурожат счетчигива жорий санаси бўлсин.Пакет танаси яратилгунга қадар TabAUD жадвалини мос равишда атрибут тоифалари яратилган бўлсин.

```
SQL> CREATE OR REPLACE PACKAGE BODY PACAA AS
        PACAA_COUNT NUMBER := 0;
        FUNCTION MULCONST(Argl NUMBER) RETURN NUMBER IS
        BEGIN
                AUDITMUL;
```

```
        RETURN Argl*PACAA_CONST;
    END;
    PROCEDURE AUDITMUL IS
    BEGIN
        PACAA_COUNT := PACAA_COUNT + 1;
        INSERT INTO TabAUD VALUES(PACAA_COUNT, SYSDATE);
        COMMIT;
    END;
END;
Package body created.
```

Пакетдан фойдаланиш

```
SQL> BEGIN
        DBMS_OUTPUT.PUT_LINE(PACAA.PACAA_CONST};
    END;
    /1.2 PL/SQL procedure successfully completed.
SQL> BEGIN
        DBMS_OUTPUT.PUT_LINE{PACAA.PACAA_COUNT);
    END;
    /DBMS_OUTPUT.PUT_LINE(PACAA.PACAA_COUNT);
    * ERROR at line 2: ORA-06550: line 2, column 30: PLS-00302: component
'PACAA_COUNT' must be declared ORA-06550: line 2, column 3:
PL/SQL: Statement ignored
SQL> BEGIN
        DBMS_OUTPUT.PUT_LINE(PACAA.MULCONST(111));
    END;
    /133.2 PL/SQL procedure successfully completed.
SQL> SELECT * FROM TabAUD;
```

Назарий саволлар:

1. *Пакетлар қандай яратилади.*
2. *Функциялар яратиш ҳақида гапириб беринг.*
3. *Яратилган пакетлардан фойдаланиш қандай амалга оширилади.*

11 – 12 – 13 – мавзу

Фойдаланувчи процедура ва функцияларини яратиш. Пакетлар. Маълумот база триггерлари ORACLE ни стандарт пакетлари. Сақланувчи процедуралар. Динамик SQL. Файлли киритиш ва чиқариш . Топшириқларни бошқариш. LOB – объектларини бошқариш. PL /SQL функцияларини SQL- ифодаларида ишлатиш.

Режа:

1. *Маълумот база триггерлари*
2. *Триггерни аниқлаш*
3. *Хатоликларни қайта ишлаш*
4. *Триггерни яратиш доир мисол*
5. *Триггерни ишлатиш*

Маълумот база триггерлари

Маълумот база триггерлари – бу процедура бўлиб, аниқ ходисалар пайдо бўлиши билан ишга тушади, ходисалар асосан маълумот жадвалларини модификациялаш (такомиллаштириш) олиб ташлаш ёки кўшиш амалларини бажариши билан боғлик бўлади. Триггерни ишга туширишни бошкарадиган ҳодиса мантикий шартлар кўринишида тасвирланади. Триггерни шартларига мос ходиса вужудга келганда, Oracle сервери автоматик равишда триггерни ишлатиб юборади.

Триггер жадвалидаги маълумотларни узгартиришни учта амалларидан Insert, Delete ва Update бири бажарилганда ишга тушади.

Триггер коди, триггерни ишга туширишни бошлаш операторларгача, еки ундан кейин бажарилиши мумкин.

Масалан, триггер фойдаланувчини амаллар бажариш хукукини текшириш учун триггер ишга тушиши керак бўлса, унда албатта амалларни

бажаргунча (Before калит сузи) бўладиган триггерни ишга тушириш керак. Агар триггер маълумотларни аудитория езувлари яратишда ишлатилса, унда триггерни амалларни бажаришдан кейин (After калит сўзи) ишга туширилаганидан фойдаланиш мумкин. Триггер коди бутун жадвал билан амал бажариши мумкин, еки амал бажарилаётган хар бир сатр билан ишлашга мослашган бўлиши мумкин. Шунга караб триггерлар операторли триггерлар еки сатрли триггерлар булинади. Операторли триггерлар бутун жадвал билан амал бажарувчи коидаларни текшириш учун ишлатилади. Сатрли триггерлар сатрларни кушишда (киритишда) бутунлигини чекланишини текшириш учун ишлатилади. Сатрли триггерлар ишга тушириш, кўшимча мантикий шарт билан ойдинлаштириш мумкин.

Триггерни аниқлаш

Oracle триггерни аниклаш оператори куйидагича синтаксисга эга:

CREATE [**OR REPLACE**] **TRIGGER** [схема_номи] триггер номи {**BEFORE** |**AFTER**} [**OR** {**INSERT**|**DELETE**|**UPDATE** [**OF** устун_номи {, устун_номи…]]}} **ON** [схема_номи.] {жадвал_номи|тасавур_номи} [**FOR EACH ROW**] [**When** шарт] PL\SQL_да_дастур_спецификацияси.

- OR REPLACE параметр эски триггер матнини шартсиз алмаштириш кераклигини англатади. Агар OR REPLACE кўрсатилмаса ва триггер аниқланса, эски триггер матнини алмаштириш бажарилмайди ва хатолик ҳақида хабар чиқарилади.

- Before ёки Alter калит сўзи триггер кодини бажаришини триггерни ишга тушишини бошлаб берувчи (инициалиация қилувчи) операторлардан олдин ёки кейин ишга тушишини билдиради.

- INSERT ,DELETE ёки UPDATE калит сўзлар триггерни ишга туширувчи конкрет операторлар. Зарур бўлмаган OR калит сўзи триггерни ишга тушрадиган қўшимча операторни улайди ON калит сўзи триггер билан боғлиқ жадвал номини беради.

- Зарурий бўлмаган FOR EACH ROW калит сўзи сатрли триггерни аниқлайди. Зарурий бўлмаган WHEN калит сўзи триггер ишга тушишини бошланиши аниқловчи ҳодиса соҳасини торайтирадиган қўшимча мантиқий шартни беради (аниқлайди).

new - процедураси (олд қўшимчаси) фақат триггерлар учун ишлатиш мумкин. INSERT ёки UPDATE командаси учун устунни янги қийматини белгилайди.

Шунингдек қайта тиклаш (UPDATE) ва олиб иашлаш (DELETE) учун: old префикс ишлатилади. Унинг маьноси UPDATE ёки DELETE командаларини бажаргунга қадар устун қийматини белгилайди.

Хатоликларни қайта ишлаш

Oracle да фавқулода вазиятларни қайта ишлашда айтиб ўтиш керак. PL/SQL дастурни нуҳсонли (авария) ҳолатда якунлаш учун

RAISE_APPLICAT1ON_ERROR процедураси ишлатилади. Унинг ёрдамида фойдаланувчи томонидан аниқланадиган 1000 гача хатоликни қайта ишлаш мумкин. Бу хатоликни номерлари диапозони 20 000 дан 20 999. RAISE_APPLICAT1ON_ERROR процедураси чақириши фавқулода (махсус) вазиятни генерация қмлишга олиб келади ва процедура дастур чақирган бажарилишини тўхтатишга олиб келади. Шу билан бирга, чақирган дастур муҳитига, хатоликни тоифаси (типини) номери ва матнли хабар чиқарилади.

Триггерни яратиш доир мисол

Масала: Агар жадвалга атрибутни киритилаётган қиймати ўрта қийматдан хаддан ташқари катталашиб кетса, ишга тушадиган триггерни яратмиз. Хаддан ташқари катталикни ўлчови сифатида, мухандислик амалиётида кенг қўлланадитган «учта сигма» қоидасини қўллаймиз. Фараз килайлик Tab1 жадвали яратилган ва тўлдирилган бўлсин

```
CREATE TABLE Tab1(At1 Number);
INSERT INTO TAB1 VALUES(1);
INSERT INTO TAB1 VALUES(3);
INSERT INTO TAB1 VALUES(5);
SQL> CREATE OR REPLACE TRIGGER TRIG_TB1
        BEFORE INSERT ON Tab1 FOR EACH ROW
        DECLARE
                StatAvg NUMBER;
                StatStd NUMBER;
                StatN NUMBER;
                BEGIN
                        SELECT COUNT(Atl), SUM(At1), STDDEV(Atl) INTO
StatN, StatAvg, StatStd FROM Tab1;
                        IF (ABS(StatAvg-StatN*(:new.At1))/(SQRT(StatN)*StatStd) >
3) THEN
                                RAISE_APPLICAT1ON_ERROR(-20002, 'Слишком
большое уклонение');
                        END IF;
                END;
/Trigger created.
```

Триггерни ишлатиш

```
SQL> INSERT INTO Tab! VALUES(4);
1 row created.
SQL> INSERT INTO Tab1 VALUES(7);
INSERT INTO Tab1 VALUES(7)
* ERROR.at line 1: ORA-20002: Слишком большое уклонение ORA-06512: at
"Ul.TRIGJTBl", line 9 ORA-04088: error during execution of trigger
'U1.TRIG_TB1‘
SQL> SELECT * FROM Tab1;
```

AT1
1
3
5
4

Назарий саволлар:

1. *Триггер қандай яратилади.*
2. *Триггер қандай ишлатилади.*
3. *Триггернинг вазифаси нималардан иборат.*

14 – МАВЗУ

МАЪЛУМОТЛАР ЯХЛИТЛИГИНИ ТАЪМИНЛАШ ВОСИТАЛАРИ. ТРАНЗАКЦИЯЛАРНИ АНИҚЛАШ ВА УНИ МББТ ДАГИ РОЛИ. ТРАНЗАКЦИЯЛАРНИ БОШЛАНИШИ ВА ТУГАШИ. ТРАНЗАКЦИЯЛАРНИ БОШҚАРИШНИ SQL ИФОДАЛАРИ (COMMIT, SAVE POINT, ROLL BACK, WORK ВА БОШҚАЛАР). ЗИДДИЯТСИЗЛИК ВА ПАРАЛЛЕЛ ҚАЙТА ИШЛАШ.

Транзакциялар

Транзакция – маълумотларни қайта ишлаш операторлар кетма-кетлиги бўлиб, маълумот базаси билан ишлайдиган бўлинмас бирлик хисобланади. Транзакция маълумот базасини бир холатдан бошқа холатга ўтказади. Агар транзакция бажариш даврида операцион тизим ёки иловани ишлашида нуқсон (носозлик) учраса, нуқсон тугатилгандан кейин маълумот базасидаги маълумотлар аввалги ўз холатига қайтади. Транзакция табиий

якунлангунча , уни "қайтариш" учун (ROLLBACK) командаси ишлатиши мумкин.

Транзакцияни фиксация қилиш ёки орқага қайтариш учун SQL- тилида COMMIT [WORK], SAVE POINT ва ROLLBACK [WORK] командалар ишлатилади.

COMMIT [WORK] – транзакцияни фиксирлайди. У куйидаларни бажаришни таъминлайди:

- ✓ *Жорий транзакция қилинган барча ўзгаришларни фиксирлайди;*
- ✓ *Транзакция тугалланади;*
- ✓ *Ушбу транзакция учун барча нуқталар йўқотилади.;*
- ✓ *Транзакция жараёнида ишлатилган барча объектлар бўшатилади.*

COMMIT [WORK] командасини ишлаш сеанси тугаганда хам бериш тавсия қилинади.

Маълумотларни ўзгартириш операторларини ўз ичига олган транзакция бажарилганда куйидаги ишлар амалга оширилади:

- ✓ *қайтариш сегментида маълумот нусхаси (копияси) яратилади;*
- ✓ *файл журналида мос ёзувлврни яратиш бажарилади;*
- ✓ *маълумот базаси буферидаги ўзгаришлар бажарилади (амалга оширилади).*

Транзакция ошкор ёки ошкормас холда тугаганда, куйидаги амаллар тўплами бажарилади:

- ✓ *транзакция фиксирланди деб белгиланади;*
- ✓ *агар файл журналидан ёзувлар маълумот базасига файлига ёзилмаган бўлса, унда уни доимий хотирага ёзиш бажарилади;*
- ✓ *тизимни блокировка қилинган объектлари бўшатилади.*

COMMIT командасида COMMENT параметрини қиймати , транзакцияни тасдиқлаш изоҳини кўрсатиш имконияти боор.

SAVEPOINT – транзакцияда қатнашган маълумот база объектларини оралиқ нусхасини сақлаб қуяди, кейинчалик, зарурат пайдо бўлса , шу нуқтага (холатга) қайтиш имкони бўлиши учун ишлатилади.

SAVEPOINT сақланиш нуқталари узун транзакцияларни майда элементларга бўлиш учун ишлатилади. Узун давомли транзакцияларда (маълумот базасида етарли катта ўзгаришлар бажарадиган транзакцияларда) қисман ўзгаришларни аниқ нуқталарда сақлаб қўйиш мақсадга мувофиқдир.Бунда транзакцияни бажаришда хатолик рўй берса, транзакция қайтаришни бошланғич холатдан эмас, балки мувафақиятли тугаган нуқтадан бошлаш имконини беради.

Команда синтаксиси қуйидагича: SAVEPOINT сақлаш_нуқтасини_номи; бунда сақлаш нуқтасини номи ссақлаш нуқтасини идентификаторидан иборат.

ROLLBACK [WORK] команда маълумот базасида транзакцияни қайтаришни, яъни ўзгаришларни бекор қилинишини англатади. Орқага қайтиш одатда иловада хато бўлганда, маълумотларни яхлитлиги бузилганда ва қолган холларда бажарилади. Уни синтаксиси қуйидагича

ROLLBACK [WORK] TO сақлаш_нуқтасини_номи;

Уни ишлатиш куйидаги амалларни бажаришга олиб келади.

- ✓ *транзакцияни бажарилиши тугатилади;*
- ✓ *жорий транзакцияда бажарилган барча ўзгаришлар бекор қилинади;*
- ✓ *транзакциядаги барча блокировкалар бекор қилинади.*

Назорат саволлари:

1. *Транзация нима?*

2. *Rollback буйруғини тушунтирб беринг.*

3. *SavePoint буйруғини тушунтириб беринг.*

15 – 16 – мавзу

Маълумотлар базасини ҳимоясини таъминлаш. Ҳисоб ёзувларини яратиш ва таҳрирлаш. Фойдаланувчиларни аутентификациялаш. Тизимли ва объектли имтиёзлар. Фойдаланувчиларни объектларга киришига рухсатни назорат қилиш. Тасаввур ва процедуралар орқали имтиёзлар тайинлаш. Маълумотларни шифрлаш орқали ҳимоялаш

Режа:

1. *Хавфсизлик*
2. *Фойдаланувчиларни идентификациялаш*
3. *Имтиёзлар*
4. *Тизимли имтиёзлар бериш*
5. *GRANT командаси*
6. *Жадваллар билан ишлаш учун имтиёз қийматлар рўйхати*
7. *Объектларга мурожат имтиёзларини бериш*
8. *Объектларга мурожат имтиёзлари GRANT командаси*

Хавфсизлик

Тизимни администрациялашни мухим омилларидан бари маълумотларни ҳавфсизлигини таъминлаш механизмидир. Бунинг учун администратор маълумот база фойдаланувчиларини яратади ва уларни аниқ объектлар устида аниқ амаллар бажариш имтиёзларини бошқаради. Фойдаланувчиларни аниқлаш ва уларга мурожатни чеклаш ORACLE тизимида операцион тизим даражасида ва МББТ даражасида хал қилинади. Иккала босқич учун хам характерли стандарт ёндошиш, администротор томонидан фойдаланувчиларни рўйхатдан ўтказиш ва имтиёзлар бариш билан белгиланади.

Имтиёз тизим томонидан таъминланувчи кондайдир белги бўлиб, у фойдаланувчига аниқ амаллар бажара олиш хуқуқини белгилайди. Имтиёзлар

(privilege) - тизимда хар хил амаллар бажаришга рухсатдир. МБ сервери хар бир фойдаланувчига имтиёзлар мажмуасини беради.

Фойдаланувчиларни идентификациялаш

ORACLE ҳар бир фойдаланувчиси махсус идентификаторга эга бўлиши керак: номга ёки кириш нуқтасига. Янги идентификаторни яратиш администратор ёки шундай хуқуққа эга бўлган фойдаланувчи томонидан амалга оширилиши мумкин. Бу CREATE USER командаси билан амалга оширилади. Янги яратилган фойдаланувчи тизим объектлари билан амал бажаришгаҳеч қандай хуқуққа эга эмас. Тизимда муваффақиятли регистрациядан ўтиш учун, администратор фойдаланувчига сеансни ташкил қилиш учун имтиёзлар бери ши керак. Бунинг учун

```
GRANT CREATE SESSION TO фойдаланувчи номи;
```

командаси ишлатилади.

Мисол: Oracle фойдаланувчин регистрация қилиш ва унга мурожат учун минимал хуқуқлар бериш.

```
SQL> CONNECT SYSTEM/MANAGER@EDUC;
     Connected.
SQL> CREATE USER U1 IDENTIFIED BY U1PSW;
     User created.
SQL> CONNECT U1/U1PSW0EDUC;
     ERROR: ORA-01017: invalid username/password; logon denied
     Warning: You are no longer connected to ORACLE.
SQL> CONNECT SYSTEM/MANAGER@EDUC;
     Connected.
SQL> GRANT CREATE SESSION TO U1;
     Grant succeeded.
SQL> CONNECT U1/U1PSW0EDUC;
     Connected.
```

Имтиёзлар

Имтиёзлар иккита синфга бўлинади:

- ✓ *тизимли имтиёзлар (system privelege);*
- ✓ *объектларга мурожат имтиёзлари(object privelege);*

Имтиёзлар МБ масштабида фойдаланувчига бирон бир амални бажариш хуқуқини берадиган имтиёзлардир.

Объектларга мурожат имтиёзлари – алохида аниқ объект устида амалалрни бажариш учун фойдаланувчига бериладиган имтиёзлар. Масалан бирорта жадвалдан танлашларни бажариш.

Фойдаланувчига тизимли имтиёзлар бериш учун GRANT командаси ишлатилади.

Тизимли имтиёзлар бериш

Тизимли имтиёзлар ORACLE сервери томонидан тизимни иккита объектига берилади: фойдаланувчилар (USER) ва ролларга (ROLE). Имтиёзларни номланган тўплами ролларни ташкил қилади. Тизимли имтиёзларни фойдаланувчига бериш учун GRANT командаси ишлатилади. GRANT командасини берувчи фойдаланувчи GRANT ANY PRIVILEGE имтиёзга эга бўлиши керак.

GRANT командаси

ORACLE да тизимли имтиёзлар берадиган оператор куйидаги синтаксисга эга.

GRANT тизимли_имтиёзлар [{,тизимли_имтиёз}...] **TO** {фойдаланувчи|**PUBLIC**} [{,фойдаланувчи}...] [**WITH ADMIN OPTION**]

Масалан: Фараз килайлик U1 фойдаланувчига тизимли имтиёз CREATE TABLE берилган бўлсин. Tab1 жадвални яратиш муваффақиятли ўтади. Tab1 жадвални U2 фойдаланувчи схемасида яратишга уриниш тизим

томонидан инкор қилинади. CREATE ANY TABLE имтиёзи берилса, бу амал муваффақиятли бажарилади. Бу холатни қуйидаги листингда кўриш мумкин.

Мисол:

```
SQL> CONNECT U1/U1PSW@EDUC;
     Connected.
SQL> CREATE TABLE Tab1(At1 NUMBER);
     Tab1e created.
SQL> CREATE TABLE U2.Tab1(Atl NUMBER);
     CREATE TABLE U2.Tab1(Atl NUMBER) * ERROR at line 1:
ORA-01031: insufficient privileges
SQL> CONNECT SYSTEM/MANAGER@EDUC;
     Connected.
SQL> GRANT CREATE ANY TABLE TO Ul;
     Grant succeeded.
SQL> CONNECT U1/U1PSW8EDUC;
     Connected.
SQL> CREATE TABLE U2.Tab1(Atl NUMBER);
     Tab1e created.
```

Жадваллар билан ишлаш учун имтиёз қийматлар рўйхати

Тизимли имтиёзлар	**Тизимли имтиёзлар бажарадиган амаллар**
CREATE ANY TABLE	Фойдаланувчи ихтиёрий маълумот база схемаси жадвал яратиш хуқуқини беради
CREATE TABLE	Фойдаланувчи ўзини маълумот база схемаси жадвал яратиш хуқуқини беради. Жадвални яратиш учун у яратиладиган соҳада жадвал соҳаси учун квота бериш имтиёзи бўлиши керак.

DROP ANY TABLE	Фойдаланувчига ихтиёрий маълумот схемасида ихтиёрий жадвални олиб ташлаш рухсат беради
ALTER ANY TABLE	Фойдаланувчи ихтиёрий маълумот база схемасида ихтиёрий жадвални ўзгартириш хуқуқини беради

Объектларга мурожат имтиёзларини бериш

Объектларга мурожат имтиёзларини ORACLE сервери томонидан тизимни иккита объектига берилади: фойдаланувчилар (USER) ва ролларга (ROLE). Имтиёзларни номланган тўплами ролларни ташкил Объектларга мурожат имтиёзларини фойдаланувчига бериш учун GRANT командаси ишлатилади. GRANT командасини берувчи фойдаланувчи ёки шу объектни эгаси бўлиши керак (ушбу объект ўзини схемасида яратилган бўлиши керак), ёки имтиёз унга WITH GRANT OPTION параметри билан узатилган, ёки GRANT ANY PRIVILEGE имтиёзга эга бўлиши керак. Объектларга мурожатни имтиёзларини аниқловчи операторлар ORACLE объектларига (жадвалларга, тасавурларга, кетма кетликларга, синонимларга, процедура, функцияларга, пакетларга ва бошқаларга) мурожатни чеклашни бошқаради

Объектларга мурожат имтиёзлари GRANT командаси

ORACLE да объектлар мурожат имтиёзларини берадиган оператор куйидаги синтаксисга эга.

GRANT{объектларга мурожат имтиёзлари|**ALL PRIVILEGES**} [устун_номи [{,устун_номи}...]][{,объектларга_мурожат_имтиёзи}...] **ON** [схема_номи.} объект_номи **TO** { фойдаланувчи|**PUBLIC**} [**WITH GRANT OPTION**]

Объектга мурожат имтиёзларини параметри қийматлар рўйхатиларини баъзилари жадвалда келтирилган.

Имтиёзлар	**Имтиёзлар рухсат берадиган амаллар**
SELECT	Бу имтиёзли фойдаланувчи мос объектдан маълумотлар танлаб (ажратиб) олиши мумкин

INSERT	Бу имтиёзли фойдаланувчи мос объектга маълумотлар қўшиши мумкин. Бу имтиёз объектларни аниқ элементига рухсат берилишини аниқлаштириши мумкин
UPDATE	Бу имтиёзли фойдаланувчи мос объектни маълумотларини модификациялаши мумкин. Бу имтиёз объектларни аниқ элементига рухсат берилишини аниқлаштириши мумкин

Мисол: Жадвал эгаси U2 фойдаланувчи, U1 фойдаланувчига Tab1 жадвалда танлаш, қўшиш ва модификациялаш учун имтиёзлар беради. Фараз қилайлик Tab1 жадвал куйидагича ташкил қилинган.

```
CREATE TABLE Tab1 (Atl Number);
```

Кўрсатиб ўтилган амалларни бажариш муваффақиятли ўтади. Лекин U1 фойдаланувчини сатрни олиб ташлаш амални бажаришга уриниши тизим томонидан бекор қилинади.

```
SQL> CONNECT U2/U2PSW@EDUC;
     Connected.
SQL> GRANT SELECT, INSERT, UPDATE ON Tab1 TO Ul;
     Grant succeeded.
SQL> CONNECT U1/U1PSW@EDUC;
     Connected.
SQL> INSERT INTO U2.Tab1 VALUES (123);
     1 row created.
SQL> SELECT * FROM U2.Tab1;
```

AT1
123

```
SQL> UPDATE U2.Tab1 SET Atl = 345;
```

```
        1 row updated.
SQL> SELECT * FROM U2.Tab1;
SQL> DELETE FROM U2.Tab1;
        DELETE FROM U2.Tab1 * ERROR at line 1: ORA-01031: insufficient
privileges
```

Назорат саволлар:

1. *Фойдаланувчиларни идентификациялаш*
2. *Имтиёзлар*
3. *Тизимли имтиёзлар бериш*
4. *GRANT командаси*

17 – 18 – мавзу

Маълумотлар базасини администрациялаш. Маълумот база администраторини асосий вазифалари (масалалари). Enterprice Managerни ишлатиш. Маълумотларни сақлаш. Маълумотларни импортлаш ва экспортлаш. МББТ га хизмат кўрсатиш, янгилаш, ва тўлдириш. Маълумот базасини захира (эҳтиёт) нусхаларини яратиш. Маълумот базасини унумдорлигини ошириш.

Режа:

1. *Маълумотлар базасини администрациялаш.*
2. *Бошқариш файли.*
3. *Enterprice Managerни ишлатиш.*
4. *Маълумотларни сақлаш, импорт ва экспорт қилиш.*
5. *МББТга хизмат кўрсатиш, янгилаш ва тўлдириш.*
6. *Маълумотлар базасини захира нусхасини яратиш.*
7. *Маълумотлар базасини унумдорлигини ошириш.*

Маълумот базасини администрациялаш

Малумот база администраторини асосий вазифаси маълумот базасини ишчи ҳолатини таъминлаш ва уни унумдорлигини ошириш. Ишчи холатини таъминлашда асосий ишларидан бири – бу маълумот базасини инсталляция қилиш. Бу масалани ечишда параметрларни тўғри танлаш, ахборотларни сақлаш воситаларини конфигурациялаш, маълумотлар базасини структурасини аниқлаш ва маълумотларни сақлаш учун соҳа ажратиш.

Администратор шунингдек базага қўшимчалар ва янгилашларни (тиклашларни) ўз вақтида ўрнатишга жавоб беради. Администраторни мухим функцияларидан бирига маълумот базасини химоясини таъминлаш

ҳам киради. У ҳар доим маълумотларни резерв нусхаларини яратиш ва уни ишончли ерда сақлашни таъминлаши керак.

Бошқариш файли

Бошқариш файли махсус иккилик файли бўлиб, у ORACLE га тизимда маълумотларни қандай сақланиши зарурлиги ни хабарлаб туради. Сукут билан ORACLE бошқариш файлини учта нусхасини яратади. Ишлаш башланганда тизим конфигурация маълумотларини шу файллардан биридан ўқийди. Агар барча нусхалар зарарланган бўлса, маълумот базаси очилмайди.

Жадвалли сохалар (TableSpace) – ORACLE компанияси мутахассисларини янгилиги ҳисобланади. У маълумот база элементларини мантиқий тўплами. Администратор сифатида жадвалли сохага хилма хил объектларни (жумладан фойдаланувчиларни) жойлаштириши мумкин. Администратор (фойдаланувчи) исталган сонда кўп жадвалли сохалар яратиши мумкин ва унда нималар сақлашни ўзи ҳал қилади. Масалан: Масалани барча жадваллари битта жадвали соҳада, қолган обьъектлари бошқа жадвалли соҳада сақлаш мумкин.

Назарий саволлар:

1. *Маълумотлар базасини администрторлаш қандай амалга ошшилади.*

АДАБИЁТЛАР

1. “Oracle Database 10g DBA Handbook”, Кевин Луни, М.: Лори -2008
2. “Oracle для профессионалов”, Том Кайт М.: DiaSoft – 2003
3. “Oracle9i The Complete Reference”, Кевин Луни, Нью Йорк, Corel Ventura – 2002
4. “SQL за 10 минут”, Бен Форта, Москва, Вильямс – 2014
5. “SQL в примерах и задачах”, И.Ф.Астахова, Москва, 2014

МУНДАРИЖА