|  |  |  |
| --- | --- | --- |
| **Unigrb** | Univerzitet u Nišu  ELEKTRONSKI FAKULTET | **logo_1960_4** |

**Interna struktura i organizacija skladišta podataka u Oracle bazi podataka**

SEMINARSKI RAD

|  |  |
| --- | --- |
| Mentor: | Student: |
|  |  |
| Prof.dr Aleksandar Stanimirović | Teodora Sekulić 1644 |

Niš, april 2024. god.

Sadržaj

[1. Uvod 3](#_Toc163059604)

[2. Baza podataka i instanca baze podataka 4](#_Toc163059605)

[3. Strukture skladištenja baza podataka 5](#_Toc163059606)

[3.1 Fizičke strukture skladištenja 5](#_Toc163059607)

[3.1.1 Oracle ASM 6](#_Toc163059608)

[3.1.2 Datoteke podataka 9](#_Toc163059609)

[3.1.3 Kontrolne datoteke 15](#_Toc163059610)

[3.1.3 Online Redo Log datoteke 18](#_Toc163059611)

[3.2 Logičke strukture skladištenja 23](#_Toc163059612)

[3.2.1 Upravljanje logičkim prostorom 25](#_Toc163059613)

[3.2.2 Blokovi podataka 28](#_Toc163059614)

[3.2.3 Ekstenzije 36](#_Toc163059615)

[3.2.4 Segmenti 37](#_Toc163059616)

[3.2.5 Tablespace 41](#_Toc163059617)

[4. Zaključak 47](#_Toc163059618)

[Literatura 48](#_Toc163059619)

# Uvod

U savremenom digitalnom svetu, baze podataka predstavljaju temeljnu infrastrukturu za efikasno skladištenje, upravljanje i pristupanje podacima. Kao ključni elementi informacionih sistema, baze podataka omogućavaju čuvanje i obrađivanje ogromne količine podataka, čime pružaju osnovu za donošenje informisanih odluka i efikasno poslovanje. Ovaj seminarski rad istražuje osnovne koncepte i strukture skladištenja baza podataka. Kroz detaljan pregled fizičkih i logičkih struktura, biće razmotreno kako se podaci organizuju i čuvaju na nivou sistema, omogućavajući stabilnost, performanse i pouzdanost baze podataka.

Najpre ćemo analizirati fizičke strukture skladištenja, uključujući Oracle ASM[[1]](#footnote-1), datoteke podataka, kontrolne datoteke i online redo log datoteke. Ovi elementi čine temeljnu infrastrukturu na kojoj se podaci čuvaju i prate promene u realnom vremenu.

Nakon toga, fokusiraćemo se na logičke strukture skladištenja, istražujući upravljanje logičkim prostorom, blokove podataka, ekstenzije, segmente i tablespace-ove. Razumevanje ovih ključnih koncepata omogućava nam sveobuhvatni uvid u način na koji se podaci organizuju i pristupaju na nivou baze podataka, što rezultira efikasnim upravljanjem resursima i optimizacijom performansi.

Kroz ovaj rad, može se steći celovit uvid u kompleksnost skladištenja baza podataka i značaj pravilne organizacije podataka za stabilnost, karakteristike i integritet sistema. Ovaj uvid predstavlja osnovu za dalje istraživanje i razumevanje složenih sistema baza podataka i njihovog skladištenja.

# Baza podataka i instanca baze podataka

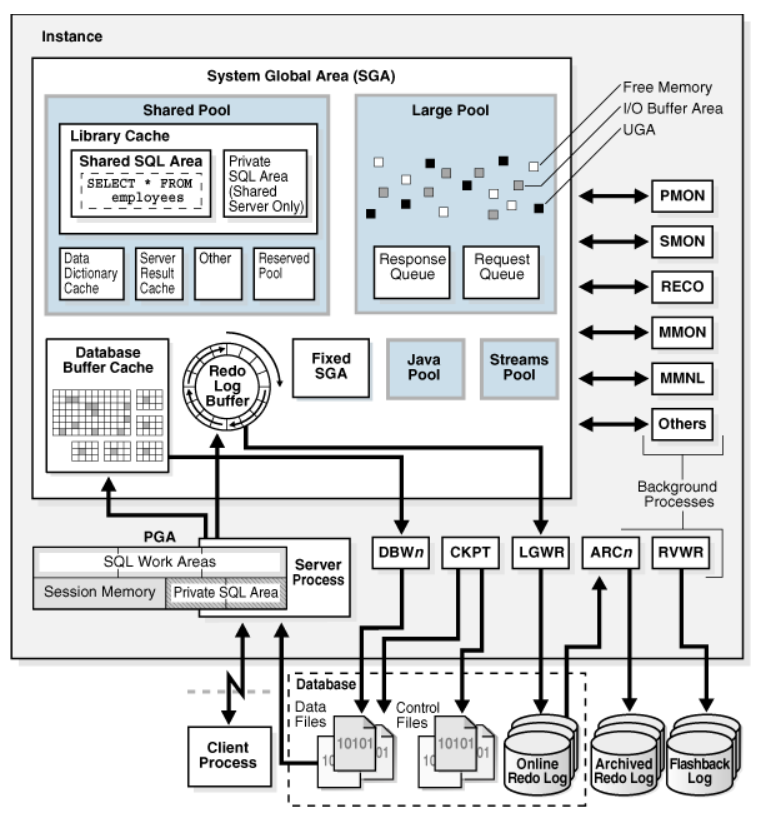
Server baze podataka je ključ za upravljanje informacijama. Uopšteno, server pouzdano upravlja velikom količinom podataka u višekorisničkom okruženju tako da korisnici mogu istovremeno pristupati istim podacima. Server baze podataka takođe sprečava neovlašćeni pristup i pruža efikasna rešenja za oporavak od kvarova.

Oracle server baze podataka sastoji se od baze podataka i barem jedne instance baze podataka, često nazvane samo instanca. Zbog toga što su instanca i baza podataka tako blisko povezani, pojam Oracle baze podataka se ponekad koristi da bi se referisalo i na instancu i na bazu podataka.

U najstrožem smislu, ovi pojmovi imaju sledeća značenja:

* **Baza podataka** je skup fajlova smeštenih na disku, koji čuvaju podatke. Ovi fajlovi mogu postojati nezavisno od instance baze podataka.
* **Instanca baze podataka** je skup memorijskih struktura koje upravljaju fajlovima baze podataka. Instanca se sastoji od zajedničkog memorijskog prostora, nazvanog sistemski globalni prostor (SGA), i skupa pozadinskih procesa. Instanca može postojati nezavisno od fajlova baze podataka.

Sledeća slika (2.1) prikazuje bazu podataka i njenu instancu. Za svaku konekciju korisnika na instancu, klijentski proces pokreće aplikaciju. Svaki klijentski proces je povezan sa svojim server procesom. Dok server proces ima svoj privatni memorijski prostor sesije, poznat kao programski globalni prostor (PGA).



**Slika 2.1** Oracle instanca i baza podataka

Slika prikazuje SGA (Sistemski globalni prostor). Unutar SGA nalaze se keš za bufere baze podataka (Database Buffer Cache), keš za ponovne zapise (Redo Log Buffer), deljani keš (Shared Pool), veliki keš (Large Pool), fiksni SGA (Fixed SGA), java keš (Java Pool) i keš za tokove (Streams Pool). Desno od SGA nalaze se pozadinski procesi PMON, SMON, RECO, MMON, MMNL i ostali. Ispod SGA nalaze se DBWn, CKPT, LGWR, ARCn i RVWR. Ispod SGA nalaze se PGA i Server Proces. Server Proces je povezan sa klijentskim procesom. Desno od klijentskog procesa nalaze se fajlovi baze podataka (Data Files, Control Files, Online Redo Log), Archived Redo Log i Flashback Log.

Iako u strogoj definiciji Oracle baza podataka predstavlja skup fizičkih struktura (fajlova i memorijskih struktura), aplikacije mogu interagovati sa više logičkih baza podataka unutar jedne fizičke baze podataka, ili jednom logičkom bazom podataka distribuiranom preko više fizičkih baza podataka.

# Strukture skladištenja baza podataka

Baza podataka može se posmatrati i sa fizičkog i sa logičkog stanovišta. Fizički podaci su podaci koji su vidljivi na nivou operativnog sistema. Na primer, alati operativnog sistema poput Linux ls i ps mogu da izlistaju fajlove baze podataka i procese. Logički podaci kao što je tabela imaju smisao samo za bazu podataka. SQL naredba može da izlista tabele u Oracle bazi podataka, ali alat operativnog sistema to ne može.

Baza podataka ima fizičke strukture i logičke strukture. Zbog toga što su fizičke i logičke strukture odvojene, moguće je upravljati fizičkim skladištenjem podataka bez uticaja na pristup logičkim strukturama skladištenja. Na primer, promena imena fizičkog fajla baze podataka ne menja imena tabela čiji su podaci smešteni u tom fajlu.

## 3.1 Fizičke strukture skladištenja

Jedna od karakteristika RDBMS-a je nezavisnost logičkih struktura podataka poput tabela, pogleda i indeksa od fizičkih struktura skladištenja.

Oracle baza podataka je skup datoteka koje čuvaju Oracle podatke na trajnom skladištu diska. Ovo poglavlje razmatra datoteke baze podataka koje se generišu nakon izdate SQL naredbe CREATE DATABASE:

* **Datoteke podataka i privremene datoteke**

Datoteka sa podacima je fizička datoteka na disku koju je kreirala Oracle baza podataka i sadrži strukture poput tabela i indeksa. Privremena datoteka je fajl sa podacima koji pripada privremenom tablespace-u. Baza podataka zapisuje podatke u ove datoteke u Oracle-ovom vlastitom formatu koji ne može biti čitan od strane drugih programa.

* **Kontrolne datoteke**

Kontrolna datoteka je osnovni fajl koji prati fizičke komponente baze podataka.

* **Online redo log datoteke**

Online redo log datoteka je set fajlova koji sadrže zapise promena napravljenih u podacima.

Postoji nekoliko mehanizama za dodeljivanje i upravljanje skladištenjem ovih fajlova. Najčešći mehanizmi uključuju:

* **Oracle automatsko upravljanje skladištem (Oracle ASM)**

Oracle ASM uključuje fajl sistem dizajniran isključivo za korišćenje od strane Oracle baze podataka.

* **Sistem fajlova operativnog sistema**

Većina Oracle baza podataka čuva fajlove u fajl sistemu, koji je struktura podataka izgrađena unutar kontinuiranog adresnog prostora diska. Svi operativni sistemi imaju menadžere fajlova koji dodeljuju i oslobađaju prostor na disku u fajlove unutar fajl sistema. Fajl sistem omogućava dodeljivanje prostora na disku mnogim fajlovima. Svaki fajl ima ime i napravljen je kao kontinuirani adresni prostor aplikacijama kao što je Oracle baza podataka. Baza podataka može kreirati, čitati, pisati, menjati veličinu i brisati fajlove. Fajl sistem se obično gradi na vrhu logičkog volumena konstruisanog od strane softverskog paketa nazvanog logički volumen menadžer (LVM). LVM omogućava da se delovi više fizičkih diskova kombinuju u jedan kontinuirani adresni prostor koji se pojavljuje kao jedan disk višim slojevima softvera.

* **Klaster fajl sistem**

Klaster fajl sistem je distribuirani sistem fajlova koji predstavlja klaster servera koji sarađuju kako bi pružili uslugu visokih performansi svojim klijentima. U Oracle RAC okruženju, klaster fajl sistem čini deljeni skladišni prostor koji se pojavljuje kao fajl sistem koji dele mnogi računari u klasterovanom okruženju. Sa klaster fajl sistemom, kvar računara u klasteru ne čini fajl sistem nedostupnim. Međutim, u operativnom sistemskom fajl sistemu, ako računar koji deli fajlove putem NFS ili drugih sredstava doživi kvar, tada je fajl sistem nedostupan.

Baza podataka koristi kombinaciju navedenih mehanizama skladištenja. Na primer, baza podataka može čuvati kontrolne fajlove i online redo log fajlove u tradicionalnom fajl sistemu, neke korisničke fajlove podataka na sirovim particijama, preostale fajlove podataka u Oracle ASM-u, i arhivirane redo log fajlove u klaster fajl sistemu.

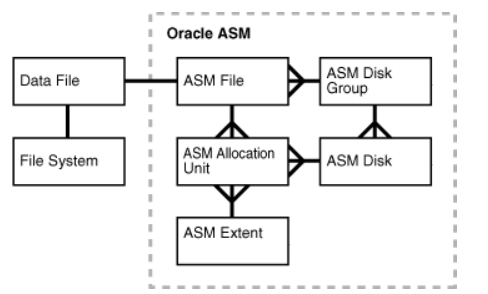
### Oracle ASM

Oracle ASM je jednostavno rešenje za skladištenje podataka Oracle baze visokih performansi. Oracle ASM je upravljač volumena i pruža fajl sistem dizajniran isključivo za korišćenje od strane baze podataka. Pruža nekoliko prednosti u odnosu na konvencionalne fajl sisteme i upravljače skladištenjem, uključujući sledeće:

* Pojednostavljuje zadatke vezane za skladište kao što su kreiranje i raspoređivanje baza podataka i upravljanje prostorom na disku
* Distribuira podatke preko fizičkih diskova kako bi eliminisao "hot spots" i obezbedio jednaku performansu preko svih diskova
* Automatski reorganizuje podatke nakon promena u konfiguraciji skladišta

Da bi koristili Oracle ASM, potrebno je dodeliti particionisane diskove za Oracle bazu podataka sa preferencijama za crtanje i preslikavanje. Oracle ASM upravlja prostorom na disku, distribuirajući I/O opterećenje na sve dostupe resurse kako bi optimizovao performanse, dok istovremeno uklanja potrebu za ručnim podešavanjem I/O operacija. Na primer, moguće je povećati veličinu diska za bazu podataka ili premestiti delove baze podataka na nove uređaje bez potrebe da se isključi baza podataka.

Oracle baza podataka može čuvati fajl sa podacima kao Oracle ASM fajl u Oracle ASM disk grupi. Unutar disk grupe, Oracle ASM pruža interfejs fajl sistema za fajlove baze podataka. Sledeća slika (3.1) prikazuje odnose između skladišnih komponenti u bazi podataka koja koristi Oracle ASM. Dijagram prikazuje odnos između Oracle ASM fajla i fajla sa podacima, iako Oracle ASM može čuvati i druge tipove fajlova.



**Slika 3.1** Komponente Oracle ASM-a

Slika 3.1 ilustruje sledeće Oracle ASM koncepte:

* **Oracle ASM diskovi**

Oracle ASM disk je skladišni uređaj koji je dodeljen Oracle ASM disk grupi. Oracle ASM disk može biti fizički disk ili particija, broj logičke jedinice (LUN) iz niza za skladištenje, logički volumen ili mrežno povezani fajl. Oracle ASM diskovi mogu biti dodavani ili uklanjani iz disk grupe dok je baza podataka u radu. Kada dodate disk u disk grupu, ili dodelite ime disku ili mu se automatski dodeli ime Oracle ASM diska.

* **Oracle ASM disk grupe**

Oracle ASM disk grupa je kolekcija Oracle ASM diskova kojima se upravlja kao logička jedinica. Podaci u disk grupi su samostalni i koriste neki prostor na disku u disk grupi. Sadržaj fajlova koji su smešteni u disk grupi je ravnomerno raspoređen, odnosno prugast, kako bi se eliminisali "hot spots" i obezbedile jednake performanse na svim diskovima.

* **Oracle ASM fajlovi**

Oracle ASM fajl je fajl smešten u Oracle ASM disk grupi. Oracle baza podataka komunicira sa Oracle ASM-om u smislu datoteka. Baza podataka može čuvati fajlove sa podacima, kontrolne fajlove, online redo log fajlove i druge tipove fajlova kao Oracle ASM fajlove. Kada baza podataka zatraži, Oracle ASM kreira Oracle ASM fajl i dodeljuje mu ime koje počinje sa znakom plus (+) nakon čega sledi ime disk grupe, na primer +DISK1.

* **Oracle ASM ekstenzija**

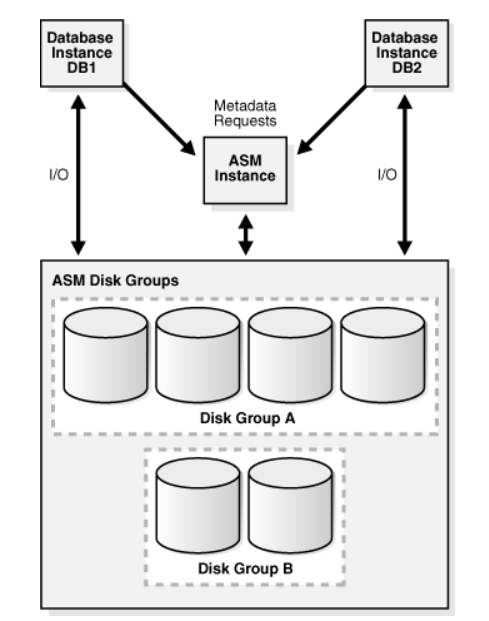
Oracle ASM ekstenzija je deo Oracle ASM fajla. Jedan Oracle ASM fajl se sastoji od jedne ili više ekstenzija. Svaka Oracle ASM ekstenzija se sastoji od jedinica alokacije na određenom disku.

* **Oracle ASM alokaciona jedinica**

Oracle ASM alokaciona jedinica je osnovna jedinica alokacije unutar disk grupe. Alokaciona jedinica je najmanji kontinuirani prostor na disku koji Oracle ASM dodeljuje. Jedna ili više alokacionih jedinica čine Oracle ASM ekstenziju.

Oracle ASM instanca je posebna Oracle instanca koja upravlja Oracle ASM diskovima. I Oracle ASM i baze podataka instance zahtevaju zajednički pristup diskovima u Oracle ASM disk grupi. Oracle ASM instance upravljaju metapodacima disk grupe i pružaju informacije o rasporedu fajlova bazi podataka instancama. Instanca baze podataka usmerava I/O ka Oracle ASM diskovima bez prolaska kroz Oracle ASM instancu. Oracle ASM instanca je izgrađena na istoj tehnologiji kao i instanca baze podataka. Na primer, Oracle ASM instanca ima sistemski globalni prostor (SGA) i pozadinske procese koji su slični onima u bazi podataka instance. Međutim, Oracle ASM instanca ne može montirati bazu podataka i obavlja manje zadatke od baze podataka instance.

Slika 3.2 prikazuje konfiguraciju jednog čvora sa jednom Oracle ASM instancom i dve baze podataka instance, pri čemu je svaka povezana sa različitom bazom podataka sa jednom instancom. Oracle ASM instanca upravlja metapodacima i pruža alokaciju prostora za Oracle ASM fajlove koji čuvaju podatke za obe baze podataka. Jedna Oracle ASM disk grupa ima četiri Oracle ASM diska, a druga ima dva diska. Obe baze podataka instance mogu pristupiti disk grupama.



**Slika 3.2** Oracle ASM instanca i instance baze podataka

### Datoteke podataka

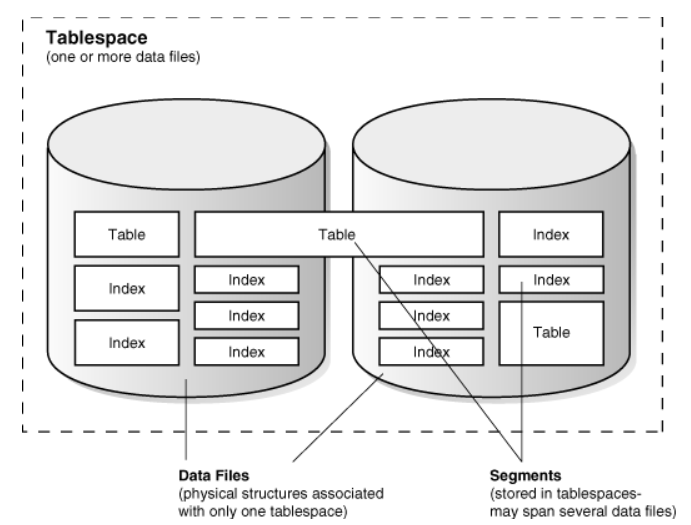
Na nivou operativnog sistema, Oracle baza podataka čuva podatke baze u strukturama nazvanim datoteke podataka (data files). Svaka Oracle baza podataka mora imati barem jednu datoteku podataka.

Oracle baza podataka fizički čuva podatke tabele u datotekama podataka.

Svaki ne-particionisani objekat šeme i svaka particija objekta čuvaju se u svom segmentu, koji pripada samo jednom tablespace-u. Na primer, podaci za ne-particionisanu tabelu čuvaju se u jednom segmentu, koji se zauzvrat čuva u jednom tablespace-u. Tablespace-ovi i datoteke podataka su tesno povezane, ali imaju važne razlike:

* Svaki tablespace se sastoji od jedne ili više datoteka podataka, koje se prilagođavaju operativnom sistemu u kojem se izvršava Oracle baza podataka.
* Podaci za bazu podataka se kolektivno čuvaju u datotekama podataka koje se nalaze u svakom tablespace-u baze podataka.
* Segment može obuhvatiti jednu ili više datoteka podataka, ali ne može obuhvatiti više tablespace-ova.
* Baza podataka mora imati tablespace-ove SYSTEM i SYSAUX. Oracle baza podataka automatski dodeljuje prve datoteke podataka bilo koje baze podataka za tablespace SYSTEM prilikom kreiranja baze podataka. Tablespace SYSTEM sadrži rečnik podataka, skup tabela koji sadrži metapodatke baze podataka. Tipično, baza podataka takođe ima undo tablespace i privremenu tablespace (obično nazvanu TEMP).

Sledeća slika (3.3) prikazuje odnos između tablespace, datoteka podataka i segmenata.



**Slika 3.3** Datoteke podataka i tablespace-ovi

**Permanentni tablespace**-ovi sadrže stalne objekte šeme. Objekti u permanentnim tablespace-ovima čuvaju se u datotekama podataka.

**Privremeni tablespace**-ovi sadrže objekte šeme samo tokom trajanja sesije. Lokalno upravljani privremeni tablespace-ovi imaju privremene datoteke (temp datoteke), koje su posebne datoteke dizajnirane za čuvanje podataka u operacijama hash, sortiranja i drugih operacija. Temp datoteke takođe čuvaju podatke rezultata kada ne postoji dovoljno prostora u memoriji. Temp datoteke su slične permanentnim datotekama podataka, sa sledećim izuzecima:

* Stalni objekti baze podataka poput tabela nikada nisu smešteni u temp datoteke.
* Temp datoteke uvek su postavljene u NOLOGGING režim, što znači da za njih nikada nije generisan redo zapis. Media recovery ne prepoznaje temp datoteke.
* Nije moguće napraviti temp datoteku samo za čitanje.
* Nije moguće kreirati temp datoteku pomoću ALTER DATABASE naredbe.
* Prilikom kreiranja ili promene veličine temp datoteka, nije uvek garantovana alokacija prostora na disku za navedenu veličinu datoteke. Na sistemima datoteka poput Linux-a i UNIX-a, temp datoteke se kreiraju kao *retke datoteke*. U ovom slučaju, disk blokovi se alociraju ne prilikom kreiranja ili promene veličine datoteke, već kada se blokovima pristupi prvi put.
* Informacije o temp datotekama prikazuju se u pogledu rečnika podataka DBA\_TEMP\_FILES i dinamičkom pogledu performansi V$TEMPFILE, ali ne i u pogledu DBA\_DATA\_FILES ili V$DATAFILE pogledu.

Svaka datoteka podataka može biti ili online ili offline. Moguće je promeniti dostupnost pojedinačnih datoteka podataka ili temp datoteka tako što će biti postavljene na offline ili vraćene na online. Baza podataka ne može pristupiti datotekama podataka koje su offline dok se ne vrate na online. Datoteke podataka je moguće staviti na offline iz mnogo razloga, uključujući izvođenje offline rezervnih kopija ili otkrivanje oštećenja bloka. Baza podataka automatski stavlja datoteku podataka na offline ako ne može da piše u nju.

Kao i datoteka podataka, i sam tablespace može biti offline ili online. Kada stavite datoteku podataka offline u online tablespace, sam tablespace ostaje online. Moguće je učiniti sve datoteke podataka jednog tablespace-a privremeno nedostupnim stavljanjem samog tablespace-a na offline.

Počevši od Oracle Database 12c verzije, moguće je koristiti ALTER DATABASE MOVE DATAFILE naredbu kako bi premestili online datoteku podataka iz jedne fizičke datoteke u drugu dok je baza podataka otvorena i pristupa datoteci. Moguće je koristiti ovu tehniku za postizanje sledećih ciljeva:

* Premeštanje tablespace-a sa jedne vrste skladišta na drugu
* Premeštanje datoteke podataka koje se retko koriste na jeftinije skladište
* Kreiranje tablespace-a samo za čitanje i premestite njene datoteke podataka na skladište za jednokratno pisanje, poput "write once read many" (WORM) drajva
* Premeštanje baze podataka u Oracle ASM

Oracle baza podataka kreira datoteku podataka za tablespace dodeljivanjem određene količine prostora na disku plus dodatni prostor za zaglavlje datoteke podataka. Operativni sistem pod kojim radi Oracle baza podataka je odgovoran za brisanje starih informacija i ovlašćenja iz datoteke pre nego što je dodeli bazi podataka.

Zaglavlje datoteke podataka sadrži metapodatke o datoteci podataka kao što su njena veličina i kontrolna SCN tačka. Svako zaglavlje sadrži apsolutni broj datoteke, koji jedinstveno identifikuje datoteku podataka unutar baze podataka, i relativni broj datoteke, koji jedinstveno identifikuje datoteku podataka unutar tabelske prostorije.

Kada Oracle baza podataka prvi put kreira datoteku podataka, alocirani prostor na disku je formatiran ali ne sadrži korisničke podatke. Međutim, baza podataka rezerviše prostor kako bi čuvala podatke za buduće segmente povezanog tablespace-a. Kako podaci rastu u tablespace-u, Oracle baza podataka koristi slobodan prostor u datotekama podataka da bi alocirala ekstenzije za segment.

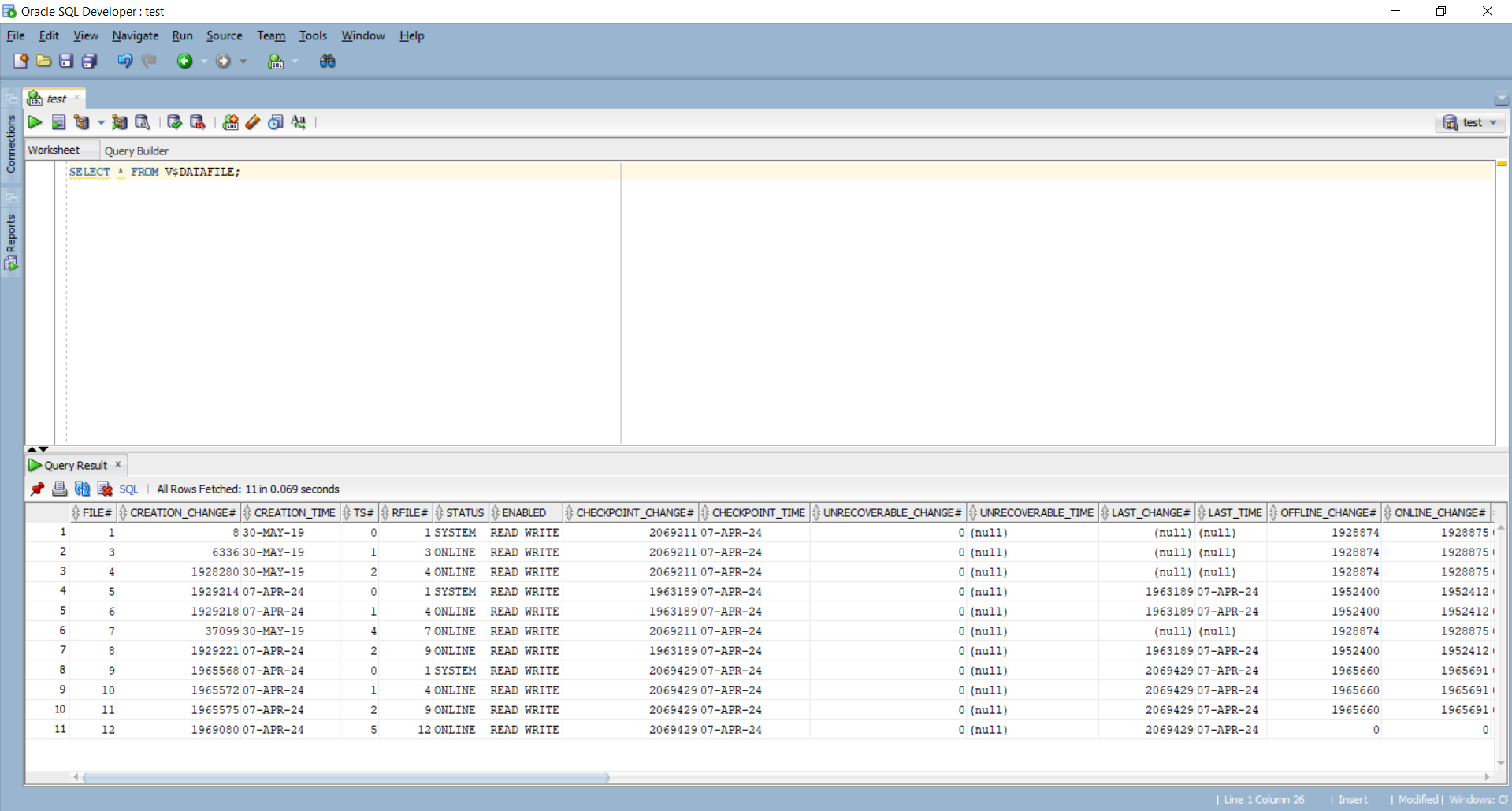
Sledeća slika (3.4) prikazuje različite tipove prostora u datoteci podataka. Ekstenzije su ili iskorišćene, što znači da sadrže podatke segmenta, ili su slobodne, što znači da su dostupne za ponovnu upotrebu. Tokom vremena, ažuriranja i brisanja objekata unutar tablespace-a mogu stvoriti prazne prostore koji pojedinačno nisu dovoljno veliki da bi se ponovno koristili za nove podatke. Ovaj tip praznog prostora naziva se *fragmentisani slobodni prostor*.

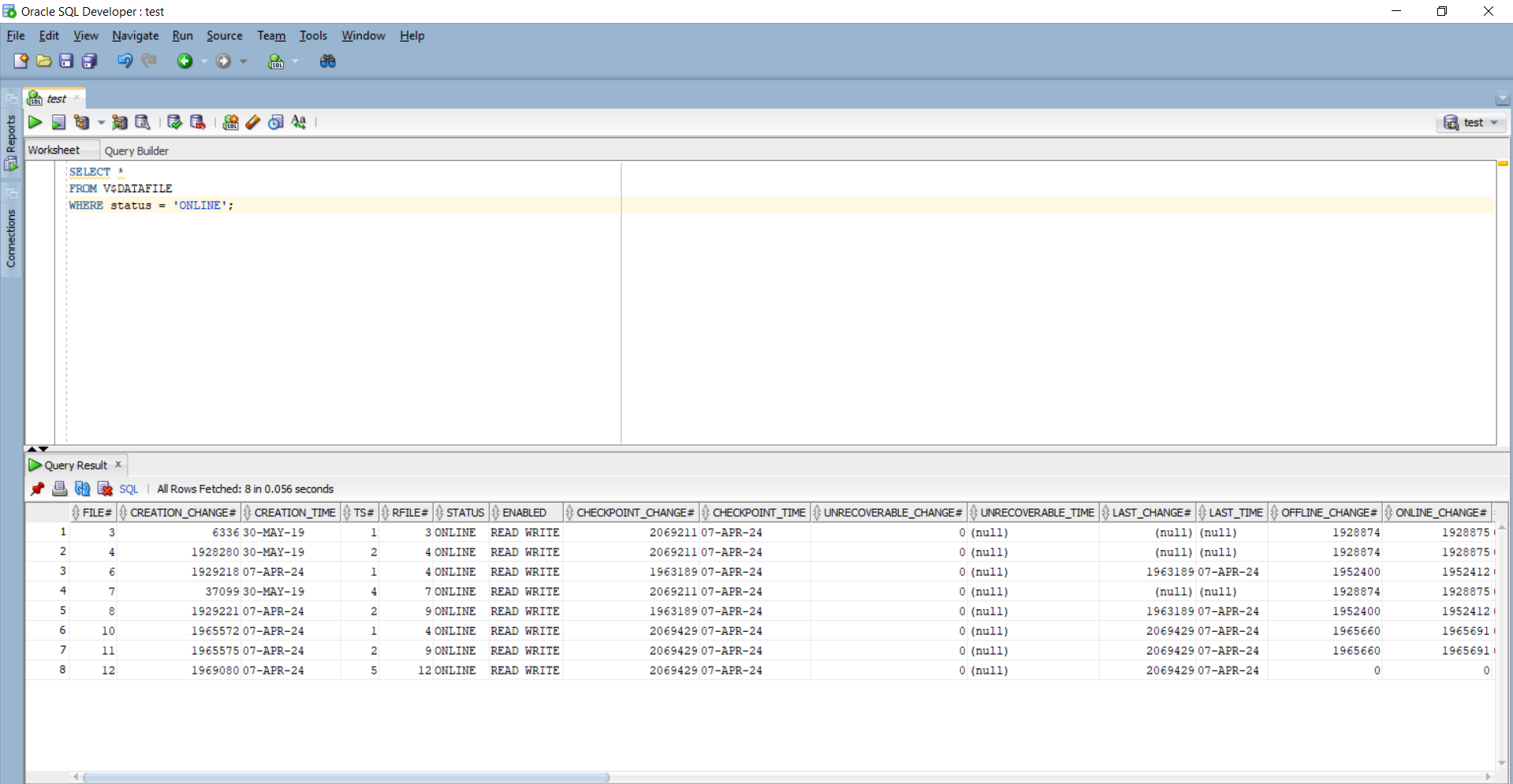


**Slika 3.4** Prostor u datoteci podataka

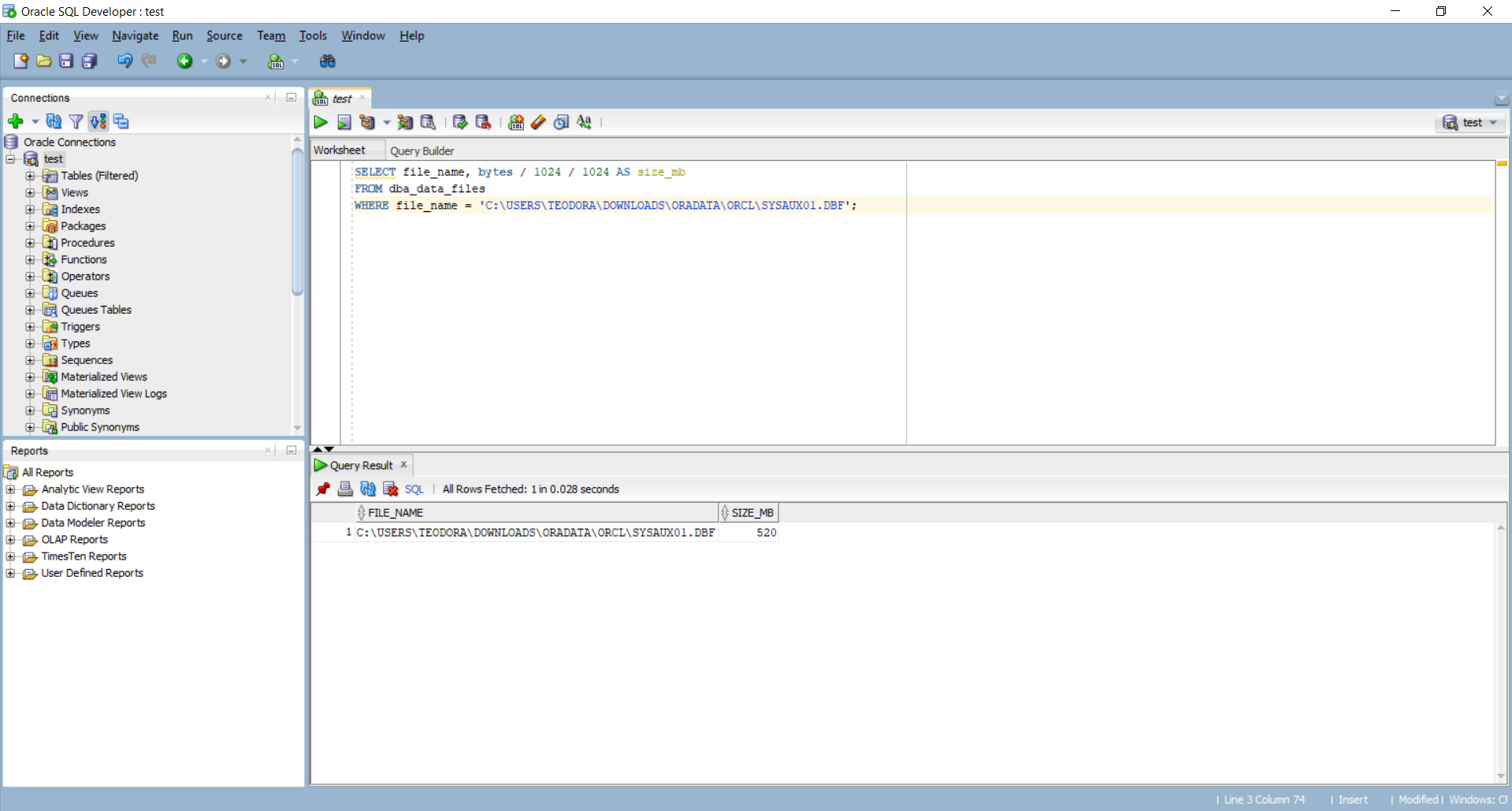
**Praktični primeri**

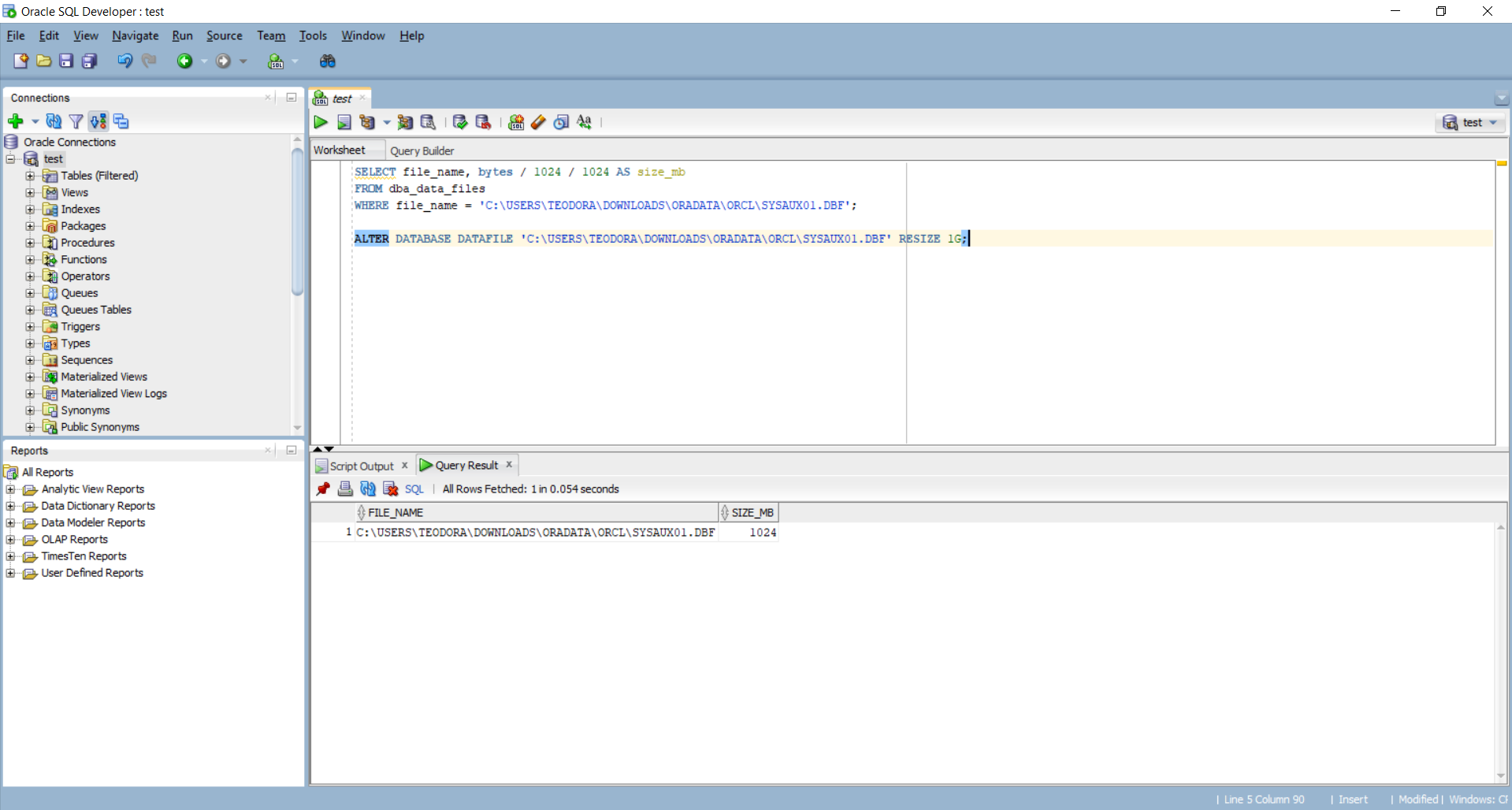
**Primer 1.** Prikaz informacija o svim datotekama podataka. I prikaz informacija o svim datotekama podataka koje su trenutno online, tj. aktivne.



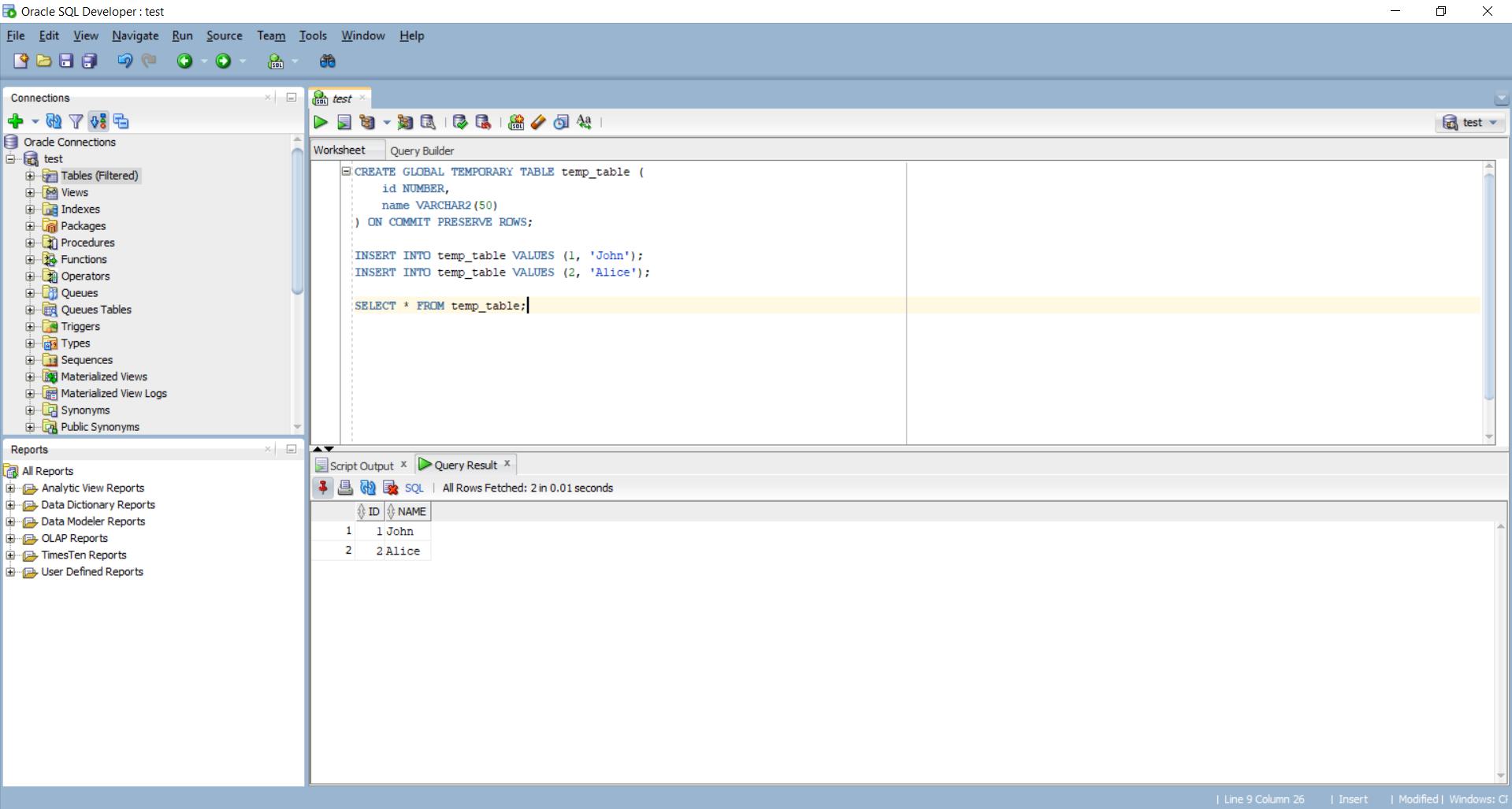


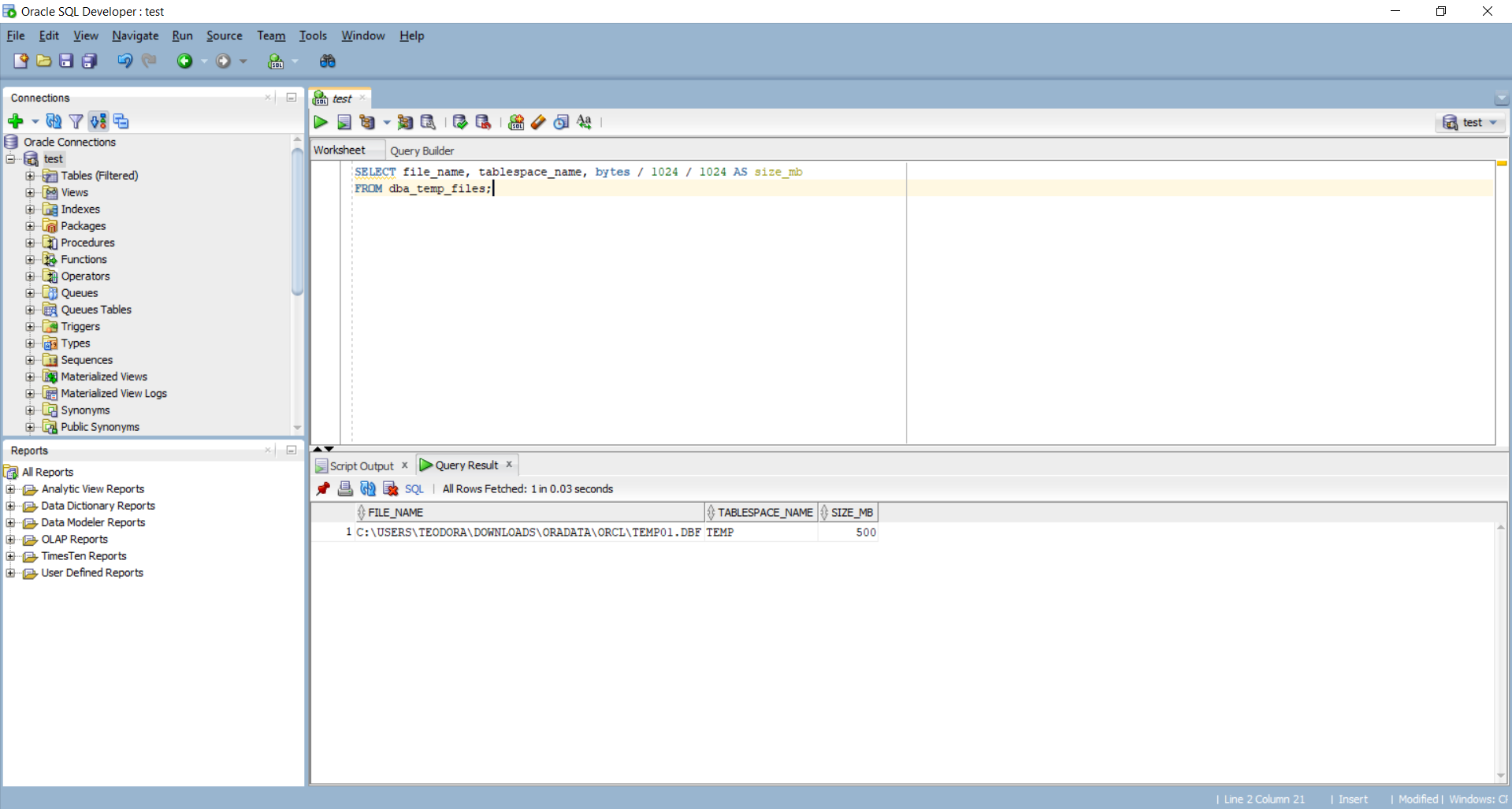
**Primer 2.** SQL upit za automatsko povećanje datoteke podataka kada je potrebno dodatno skladištenje podataka. Gde 'C:\USERS\TEODORA\DOWNLOADS\ORADATA\ORCL\SYSAUX01.DBF' predstavlja putanju do patoteke podatakaza kojoj želimo omogućiti automatsko proširivanje, 1G je veličina proširenja u gigabajtima (GB).

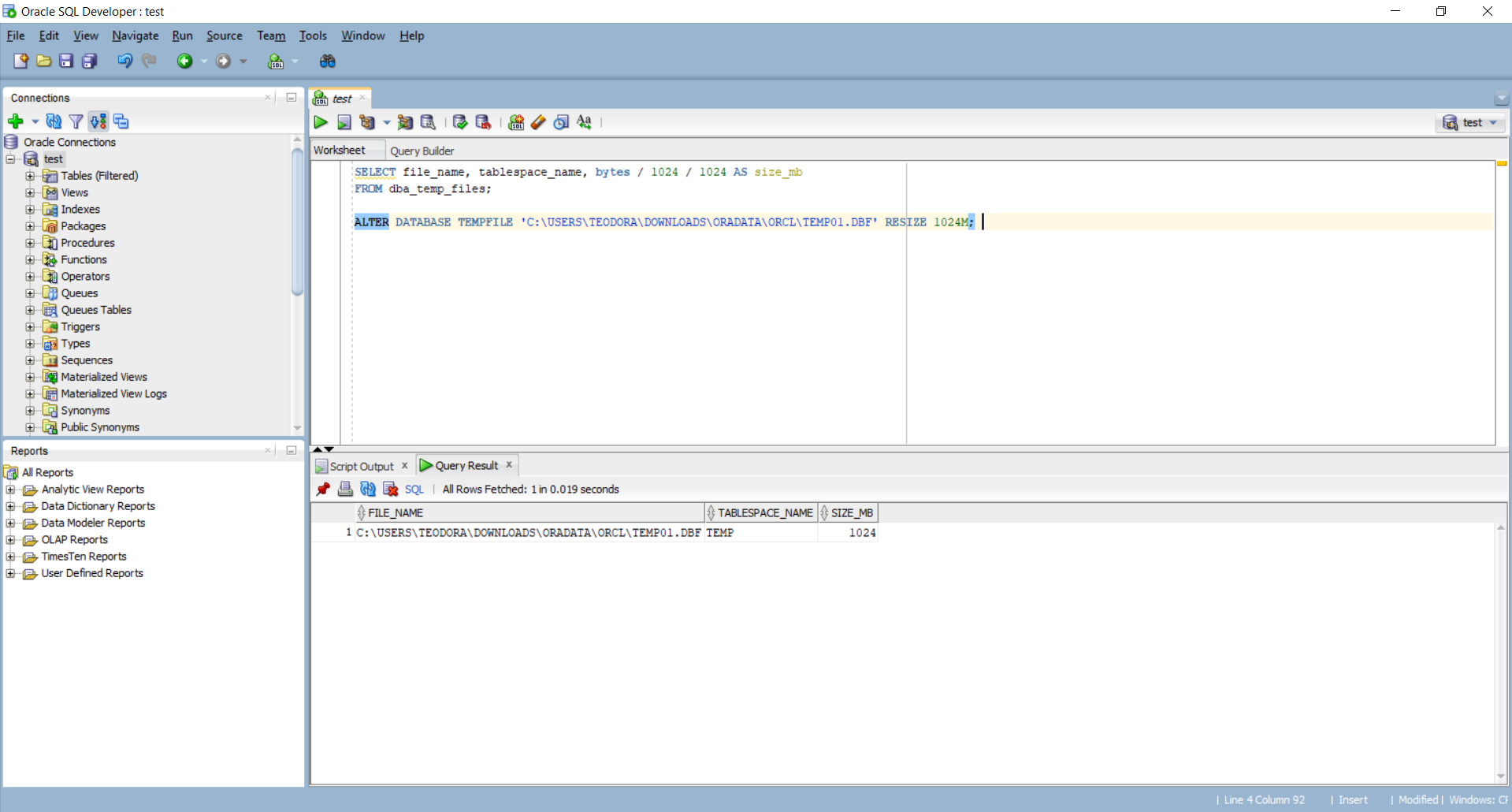




**Primer 3.** Sledće naredbe obavljaju nekoliko radnji u Oracle bazi podataka. Prvo, CREATE GLOBAL TEMPORARY TABLE se koristi za kreiranje globalne privremene tabele temp\_table sa definisanim kolonama. Zatim, INSERT INTO temp\_table VALUES ... naredbe ubacuju podatke u ovu privremenu tabelu. Nakon toga, SELECT file\_name, tablespace\_name, bytes / 1024 / 1024 AS size\_mb FROM dba\_temp\_files naredba prikazuje informacije o temp fajlovima u bazi podataka, uključujući njihova imena, tablespace-ove i veličinu. Konačno, ALTER DATABASE TEMPFILE ... RESIZE 1024M naredba menja veličinu određenog temp fajla na 500 megabajta.







### 3.1.3 Kontrolne datoteke

Kontrolna datoteka baze podataka je mala binarna datoteka povezana samo s jednom bazom podataka. Svaka baza podataka ima jedinstvenu kontrolnu datoteku, iako su dozvoljene višestruke identične kopije. Oracle baza podataka koristi kontrolnu datoteku da bi pronašla datoteke baze podataka i upravljala stanjem baze podataka uopšteno.

Kontrolna datoteka sadrži informacije kao što su:

* Ime baze podataka i jedinstveni identifikator baze podataka (DBID)
* Vremenska oznaka kreiranja baze podataka
* Informacije o datotekama podataka, online redo log datotekama i arhiviranim redo log datotekama
* Informacije o tablespace-ovima
* RMAN [[2]](#footnote-2)rezervne kopije

Kontrolna datoteka ima sledeće svrhe:

* Sadrži informacije o datotekama podataka, online redo log datotekama i drugim elementima potrebnim za otvaranje baze podataka.
* Kontrolna datoteka prati strukturne promene u bazi podataka. Na primer, kada administrator dodaje, menja ime ili briše datoteku podataka ili online redo log datoteku, baza podataka ažurira kontrolnu datoteku da bi odražavala ovu promenu.
* Sadrži metapodatke koji moraju biti dostupni kada baza podataka nije otvorena. Na primer, kontrolna datoteka sadrži informacije potrebne za oporavak baze podataka, uključujući kontrolne tačke (checkpoints). Kontrolna tačka označava SCN u redo toku gde bi bilo potrebno započeti oporavak instance. Svaka potvrđena promena pre kontrolne tačke SCN je garantovana da je sačuvana na disku u datotekama podataka. Najmanje svakih tri sekunde, proces kontrolne tačke beleži informacije u kontrolnoj datoteci o poziciji kontrolne tačke u online redo log datoteci.

Oracle baza podataka neprekidno čita i piše u kontrolnu datoteku tokom korišćenja baze podataka i mora biti dostupna za pisanje kad god je baza podataka otvorena. Na primer, oporavak baze podataka uključuje čitanje imena svih datoteka podataka sadržanih u bazi podataka iz kontrolne datoteke. Druge operacije, poput dodavanja datoteke podataka, ažuriraju informacije koje su sačuvane u kontrolnoj datoteci.

Oracle baza podataka omogućava otvaranje i pisanje višestrukih, identičnih kontrolnih datoteka istovremeno u istoj bazi podataka. Replikacija kontrolne datoteke na različitim diskovima omogućava bazi podataka da postigne redundantnost i na taj način izbegne jednu tačku kvara.

Ako kontrolna datoteka postane neupotrebljiva, tada instanca baze podataka ne može da se pokrene kada pokuša da pristupi oštećenoj kontrolnoj datoteci. Kada postoje druge trenutne kopije kontrolne datoteke, moguće je ponovo montirati bazu podataka i otvoriti je bez oporavka medija. Međutim, ako su sve kontrolne datoteke baze podataka izgubljene, instanca baze podataka se gasi i neophodan je oporavak medija. Oporavak medija nije jednostavan ako se mora koristiti starija rezervna kopija kontrolne datoteke zato što trenutna kopija nije dostupna.

Informacije o bazi podataka čuvaju se u različitim sekcijama kontrolne datoteke. Svaka sekcija je skup zapisa o određenom aspektu baze podataka. Na primer, jedna sekcija u kontrolnoj datoteci prati datoteke podataka i sadrži skup zapisa, jedan za svaku datoteku podataka. Svaka sekcija je smeštena u više logičkih blokova kontrolne datoteke. Zapisi mogu obuhvatati blokove unutar sekcije.

Kontrolna datoteka sadrži sledeće vrste zapisa:

* **Zapisi sa cirkularnim ponovnim korišćenjem**

Zapis sa cirkularnim ponovnim korišćenjem sadrži nekritične informacije koje su spremne da se prepišu ako je potrebno. Kada su svi dostupni slotovi zapisa popunjeni, baza podataka ili proširuje kontrolnu datoteku da napravi mesto za novi zapis ili prepisuje najstariji zapis. Primeri uključuju zapise o arhiviranim redo log datotekama i RMAN rezervnim kopijama.

* **Zapisi sa ne-cirkularnim ponovnim korišćenjem**

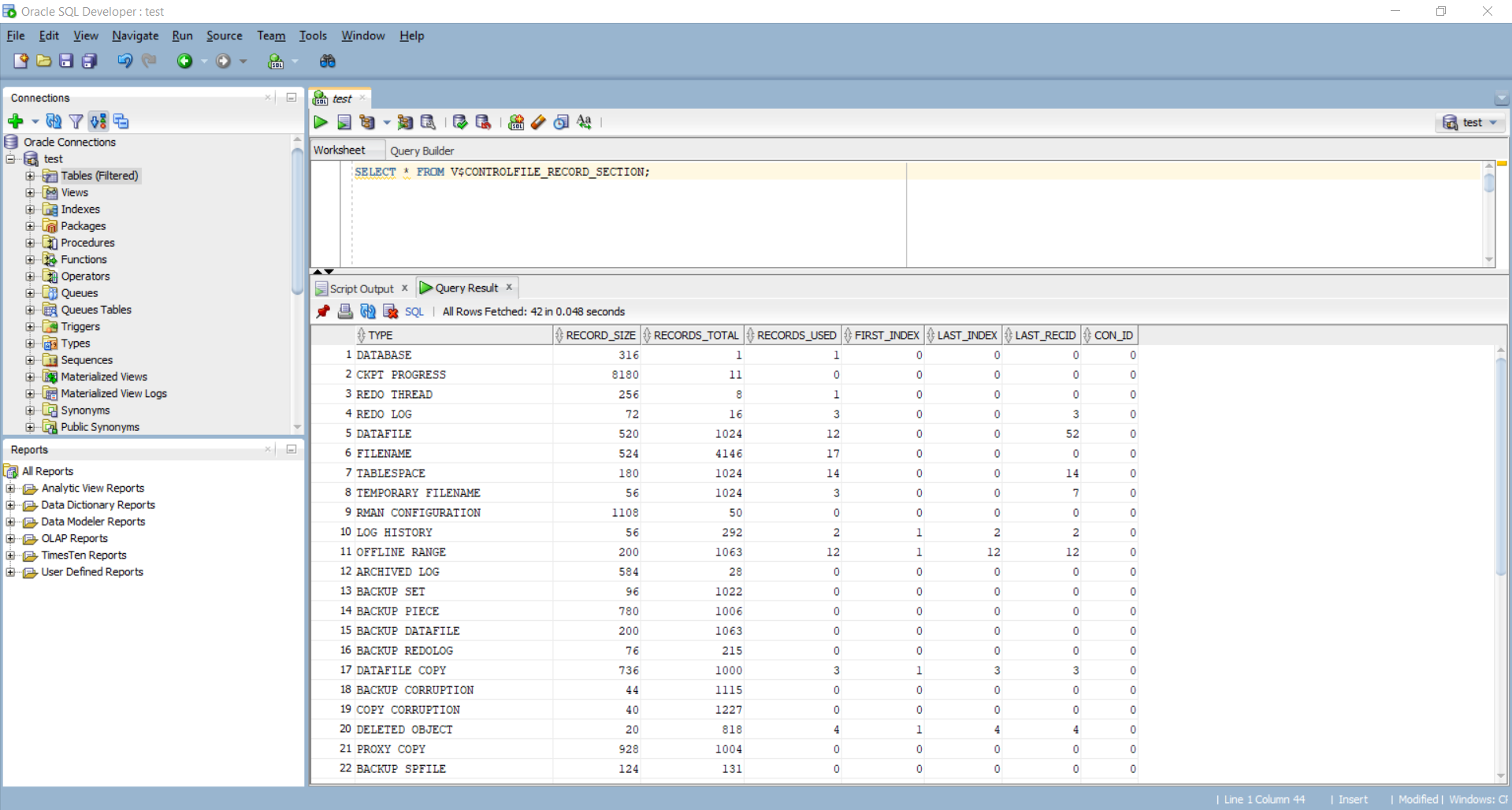
Zapis sa ne-cirkularnim ponovnim korišćenjem sadrži kritične informacije koje se retko menjaju i ne mogu se prepisati. Primeri informacija uključuju tablespace-ove, datoteke podataka, online redo log datoteke i redo niti. Oracle baza podataka nikada ne koristi ponovo ove zapise osim ako odgovarajući objekat nije uklonjen iz tablespace-a.

Moguće je izvršiti upit dinamičkih performantnih pregleda, takođe poznatih kao V$ pregledi, kako bi videli informacije sačuvane u kontrolnoj datoteci. Na primer, može se koristiti upit V$DATABASE kako bi dobili ime baze podataka i DBID. Međutim, samo baza podataka može menjati informacije u kontrolnoj datoteci.

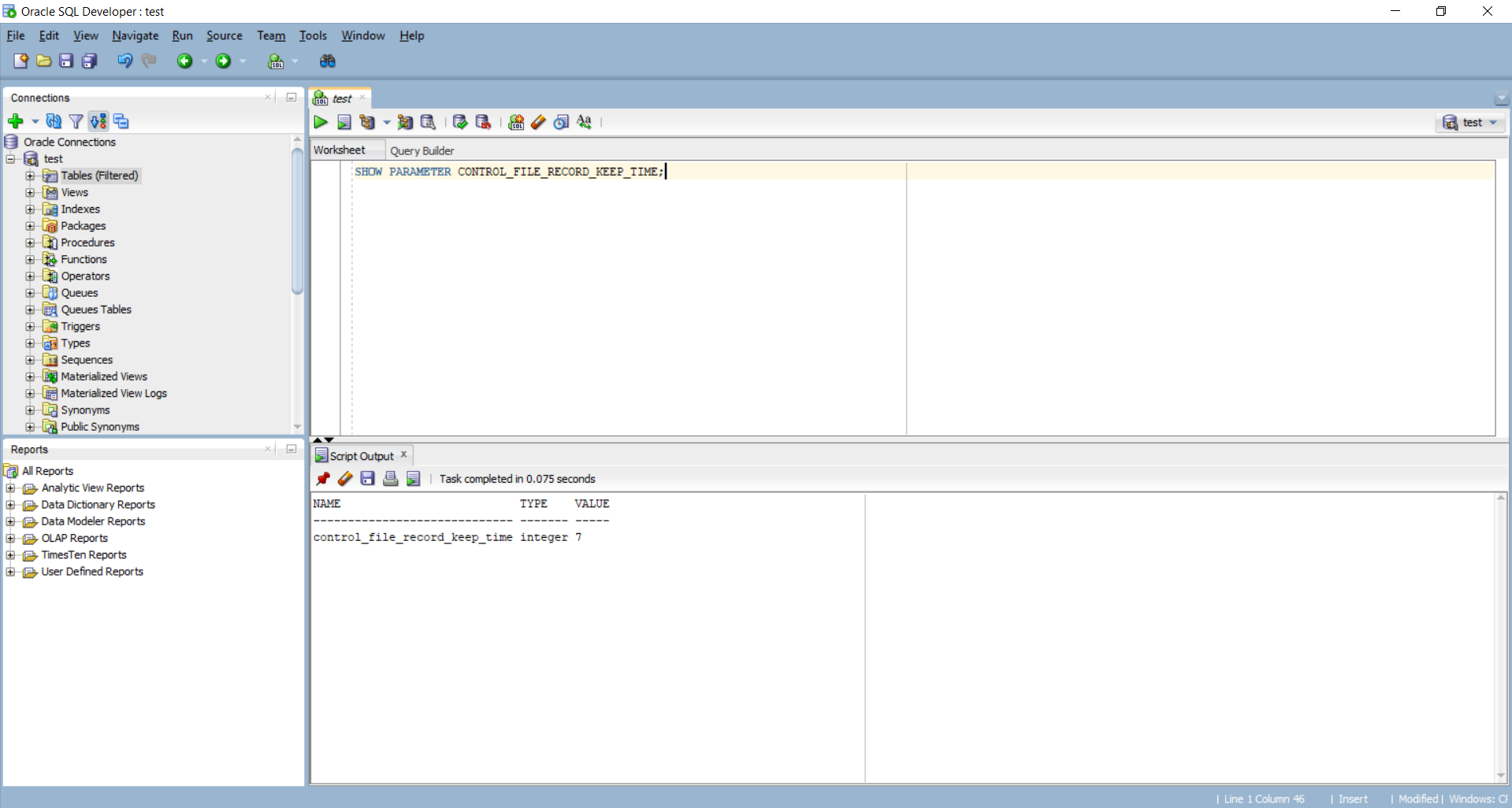
Čitanje i pisanje blokova kontrolne datoteke razlikuje se od čitanja i pisanja blokova podataka. Za kontrolnu datoteku, Oracle baza podataka čita i piše direktno sa diska u globalnu oblast programa (PGA). Svaki proces dodeljuje određenu količinu svoje PGA memorije za blokove kontrolne datoteke.

**Praktični primeri**

**Primer 4.** Pregled sekcija unutar kontrolnih fajlova.



**Primer 5.** Sledeća naredba prikazuje vrednost parametra CONTROL\_FILE\_RECORD\_KEEP\_TIME koja određuje koliko dugo će Oracle baza podataka čuvati informacije o zapisima o kontrolnim datotekama pre nego što ih automatski izbriše.



### Online Redo Log datoteke

Najvažnija struktura za oporavak je online redo log, koji se sastoji od dva ili više unapred alociranih datoteka koje čuvaju promene u bazi podataka kako se dešavaju. Online redo log beleži promene u datotekama podataka. Baza podataka održava online redo log datoteke kako bi se zaštitila od gubitka podataka. Konkretno, nakon neuspeha instance, online redo log datoteke omogućavaju Oracle bazi podataka da oporavi potvrđene podatke koje još nije upisala u datoteke podataka.

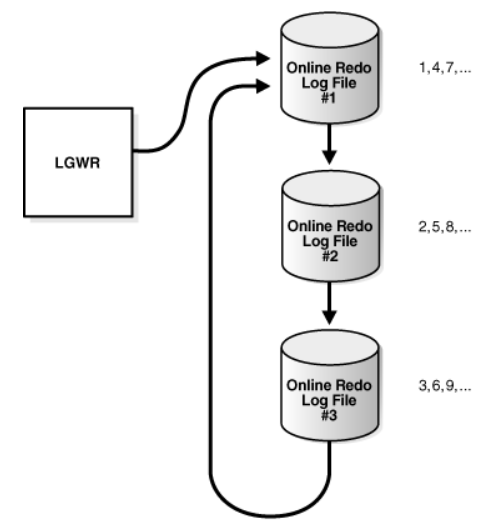
Server procesi sinhrono pišu svaku transakciju u redo log bafer, koji zatim proces LGWR piše u online redo log. Sadržaj online redo loga uključuje nepotvrđene transakcije, kao i naredbe za upravljanje šemom i objektima.

Kako baza podataka vrši promene u undo segmentima, takođe piše ove promene u online redo logove. Kao rezultat, online redo log uvek sadrži undo podatke za stalne objekte. Moguće je konfigurisati bazu podataka da čuva sve undo podatke za privremene objekte u privremenom undo segmentu, što štedi prostor i poboljšava performanse, ili je moguće dozvoliti bazi podataka da čuva i undo podatke za stalne i privremene objekte u online redo logu.

Oracle baza podataka koristi online redo log samo za oporavak. Međutim, administratori mogu upitati online redo log datoteke putem SQL interfejsa u Oracle LogMiner alatu. Redo log datoteke su korisna izvorna informacija o istorijskim aktivnostima baze podataka.

Online redo log za instancu baze podataka naziva se **redo nit**. U konfiguracijama sa jedinstvenom instancom, samo jedna instanca pristupa bazi podataka, pa je prisutna samo jedna redo nit. Međutim, u Oracle Real Application Clusters (Oracle RAC) konfiguraciji, više instanci istovremeno pristupa bazi podataka, pri čemu svaka instanca ima svoju redo nit. Odvojena redo nit za svaku instancu izbegava sukob za jedan set online redo log datoteka. Online redo log se sastoji od dve ili više online redo log datoteka. Oracle baza podataka zahteva minimum dve datoteke da bi garantovala da je uvek jedna datoteka dostupna za pisanje u slučaju da je druga datoteka u procesu brisanja ili arhiviranja.

Oracle baza podataka koristi samo jednu online redo log datoteku u jednom trenutku da bi sačuvala zapise koji su upisani iz redo log bafera. Online redo log datoteka u koju proces log pisca (LGWR) aktivno piše naziva se trenutnom online redo log datotekom. Prebacivanje loga se dešava kada baza podataka prestane da piše u jednu online redo log datoteku i počne da piše u drugu. Obično se prebacivanje dešava kada je trenutna online redo log datoteka puna i pisanje mora da se nastavi. Međutim, moguće je konfigurisati prebacivanje loga da se dešava u redovnim intervalima, bez obzira da li je trenutna online redo log datoteka popunjena, i možete ručno izazvati prebacivanje loga. Pisac loga piše u online redo log datoteke ciklično. Kada pisac loga ispuni poslednju dostupnu online redo log datoteku, proces piše u prvu log datoteku, počinjući ciklus ponovo. Slika 3.5 prikazuje ciklično pisanje redo loga.



**Slika 3.5** Ponovna upotreba Online Redo Log datoteka

Brojevi na slici 3.5 pokazuju sekvencu u kojoj LGWR piše u svaku online redo log datoteku. Baza podataka dodeljuje svakoj datoteci novi broj zapisa loga kada se dogodi prebacivanje loga i pisac loga počne da piše u nju. Kada baza podataka ponovo koristi online redo log datoteku, ova datoteka dobija sledeći dostupan broj zapisa loga.

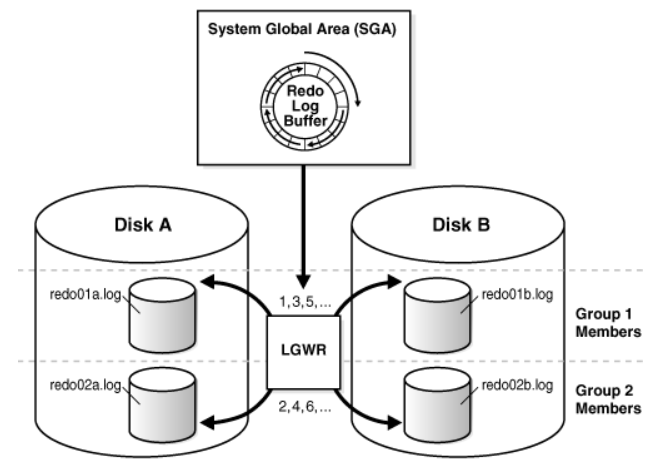
Popunjene online redo log datoteke su dostupne za ponovnu upotrebu u zavisnosti od režima arhiviranja:

* Ako je arhiviranje onemogućeno, što znači da je baza podataka u NOARCHIVELOG režimu, tada je popunjena online redo log datoteka dostupna nakon što su promene koje su zabeležene u njoj checkpointed (upisane) na disk od strane procesa upisivača baze podataka (DBW).
* Ako je arhiviranje omogućeno, što znači da je baza podataka u ARCHIVELOG režimu, tada je popunjena online redo log datoteka dostupna pisacu loga nakon što su promene upisane u datoteke podataka i datoteka je arhivirana.

U nekim okolnostima, pisac loga može biti sprečen da ponovo koristi postojeću online redo log datoteku. Aktivna online redo log datoteka je potrebna za oporavak instance, dok neaktivna online redo log datoteka nije potrebna za oporavak instance. Takođe, online redo log datoteka može biti u procesu brisanja.

Oracle baza podataka može automatski održavati dve ili više identičnih kopija online redo log datoteka na različitim lokacijama. Online redo log grupa se sastoji od online redo log datoteke i njenih redundantnih kopija. Svaka identična kopija je član online redo log grupe. Svaka grupa je definisana brojem, kao što su grupa 1, grupa 2, i tako dalje. Održavanje više članova online redo log grupe štiti od gubitka redo loga. Idealno, lokacije članova treba da budu na različitim diskovima tako da neuspeh jednog diska ne izazove gubitak cele online redo log datoteke.

Na slici 3.6, A\_LOG1 i B\_LOG1 su identični članovi grupe 1, dok su A\_LOG2 i B\_LOG2 identični članovi grupe 2. Svaki član u grupi mora biti iste veličine. LGWR piše istovremeno u grupu 1 (članovi A\_LOG1 i B\_LOG1), zatim piše istovremeno u grupu 2 (članovi A\_LOG2 i B\_LOG2), zatim piše u grupu 1, i tako dalje. LGWR nikada ne piše istovremeno u članove različitih grupa.



**Slika 3.6** Višestruke kopije Online Redo Log datoteka

**Arhivirana redo log datoteka** je kopija popunjene članice grupe online redo loga. Datoteka se ne smatra delom baze podataka, već je offline kopija online redo log datoteke kreirane od strane baze podataka i upisane na lokaciju koju je korisnik odredio.

Arhivirane redo log datoteke su ključni deo strategije za rezervne kopije i oporavak. Moguće ih je koristiti za:

* Oporavak rezervne kopije baze podataka
* Ažuriranje “standby” baze podataka
* Dobijanje informacija o istoriji baze podataka koristeći Oracle LogMiner alat

Operacija generisanja arhivirane redo log datoteke poznata je kao arhiviranje. Ova operacija može biti automatska ili ručna. Moguća je samo kada je baza podataka u ARCHIVELOG režimu. Arhivirana redo log datoteka uključuje redo unose i broj zapisa loga identičnog člana grupe online redo loga.

Na primer, datoteke A\_LOG1 i B\_LOG1 su identični članovi Grupe 1. Ako je baza podataka u ARCHIVELOG režimu, i ako je automatsko arhiviranje omogućeno, tada će proces arhivera (ARCn) arhivirati jednu od ovih datoteka. Ako je A\_LOG1 oštećen, tada proces može arhivirati B\_LOG1. Arhivirana redo log datoteka sadrži kopiju svake grupe redo logova koja je postojala od trenutka kada ste omogućili arhiviranje.

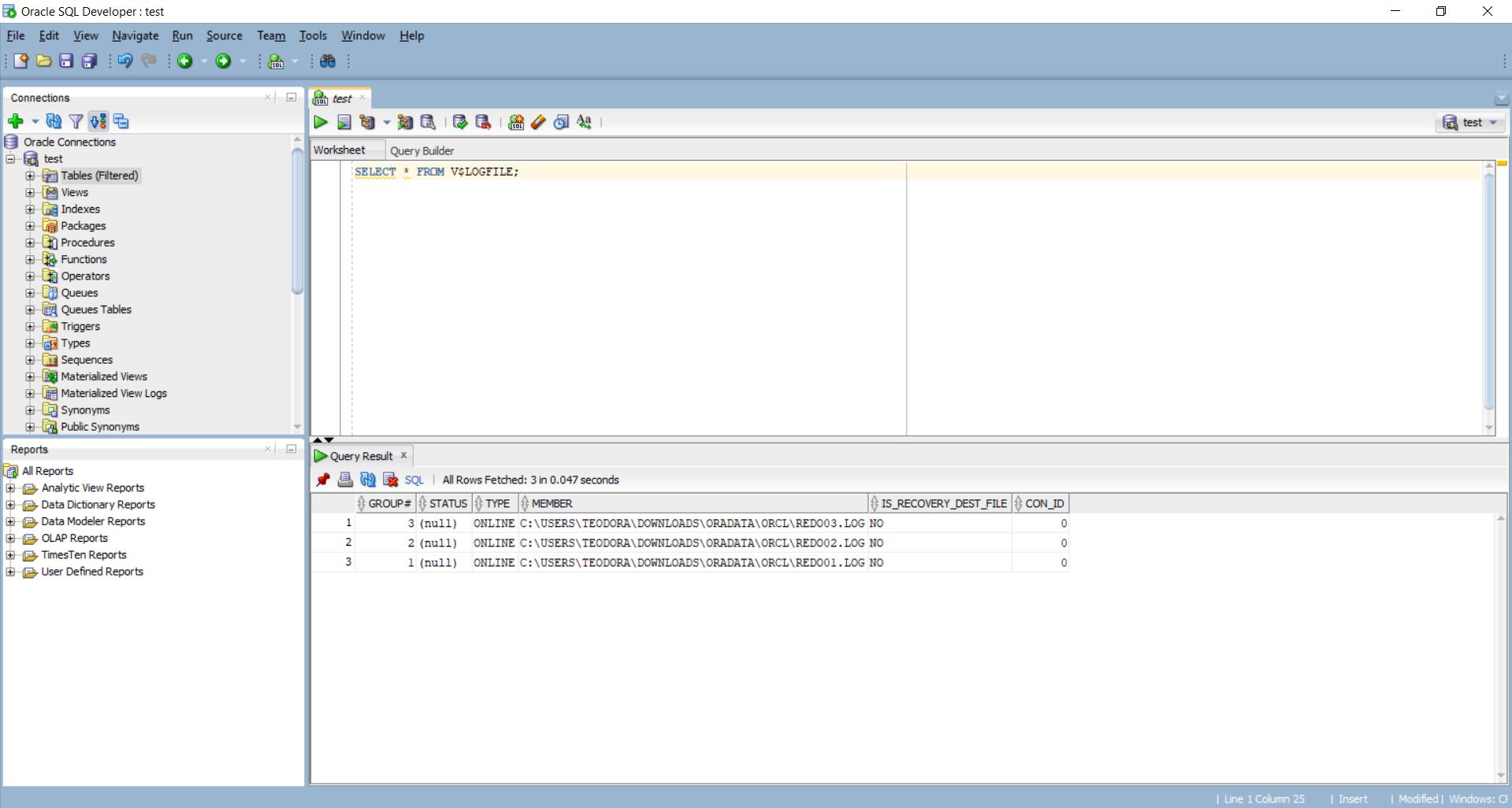
Online redo log datoteke sadrže **redo zapise**. Redo zapis se sastoji od grupe vektora promena, pri čemu svaki vektor opisuje promenu na bloku podataka. Na primer, ažuriranje plate u tabeli zaposlenih generiše redo zapis koji opisuje promene na bloku podataka segmenta tabele, bloku podataka undo segmenta i tabeli transakcija undo segmenata.

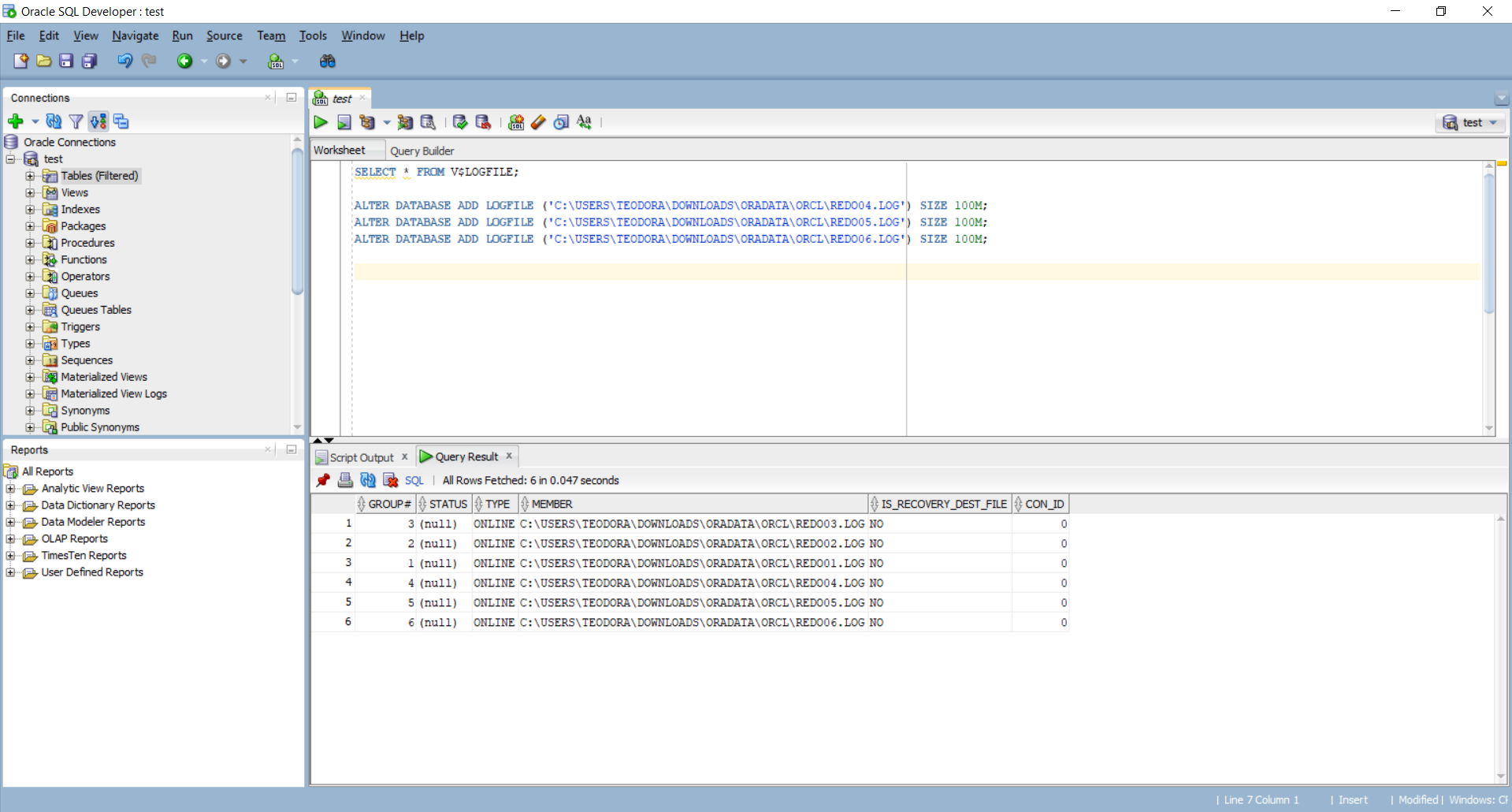
Redo zapisi sadrže sve relevantne metapodatke za promenu, uključujući sledeće:

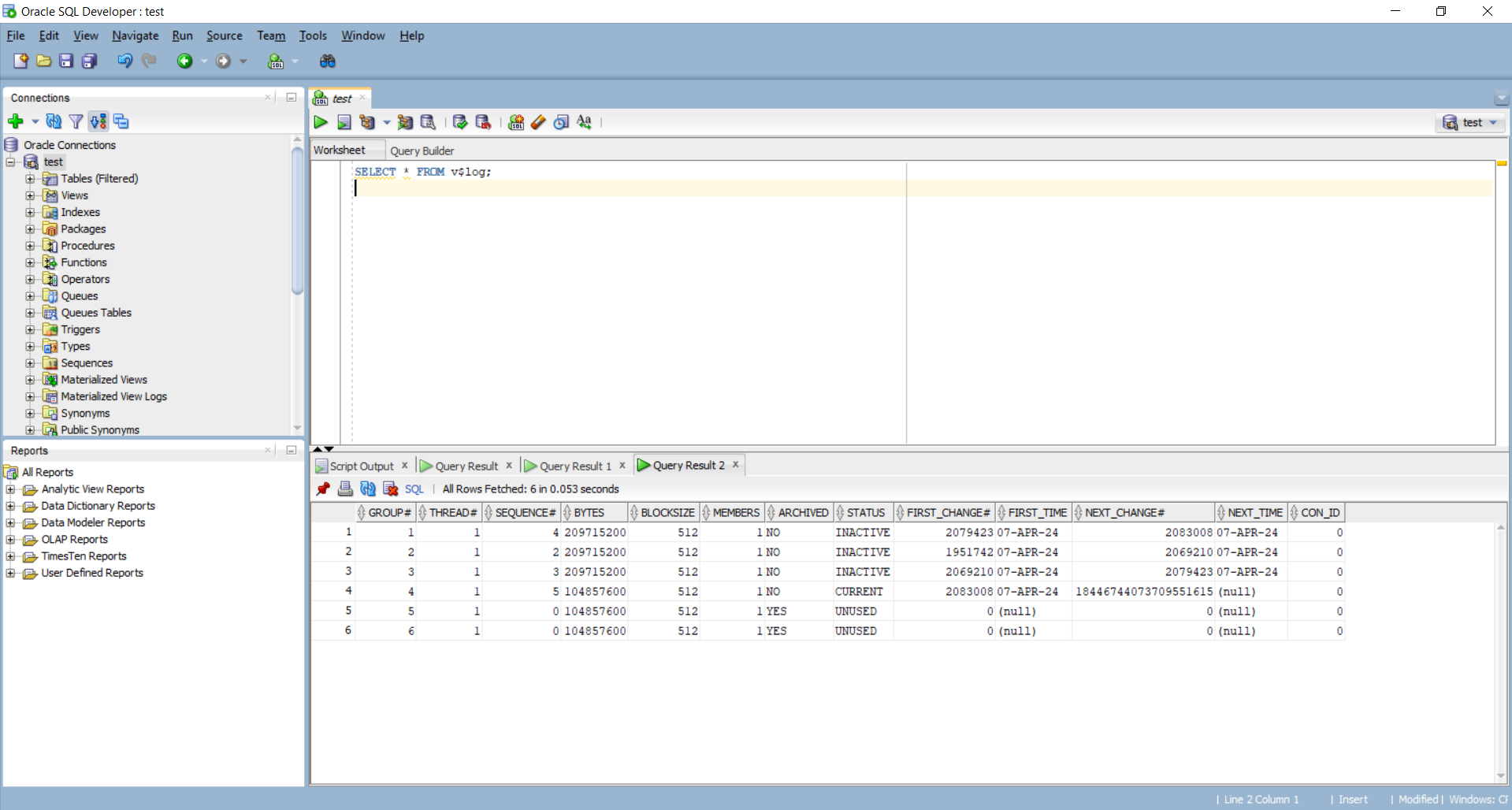
* SCN i vremensku oznaku promene
* ID transakcije koja je generisala promenu
* SCN i vremensku oznaku kada je transakcija potvrđena (ako je potvrđena)
* Vrsta operacije koja je napravila promenu
* Ime i tip modifikovanog segmenta podataka

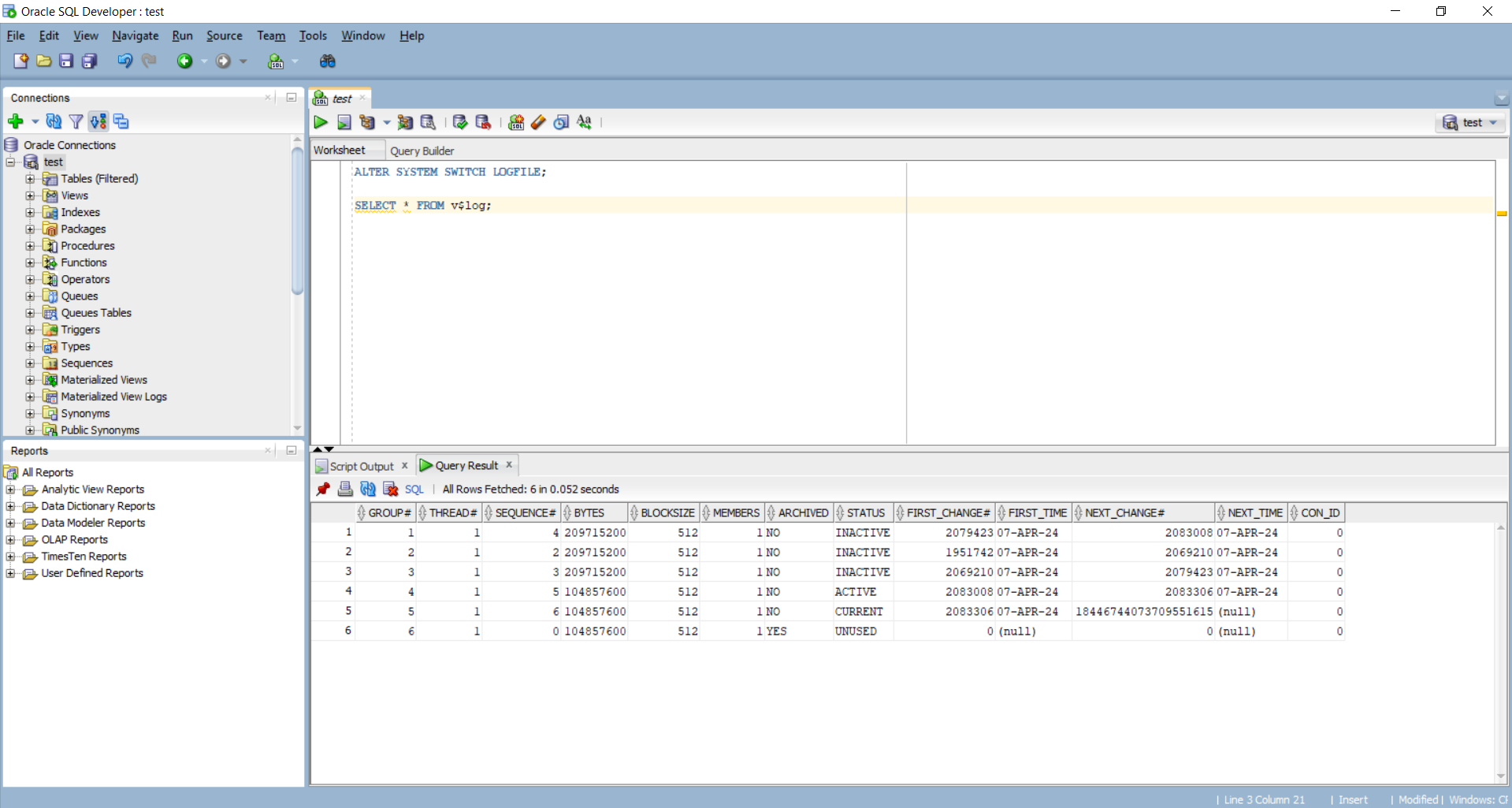
**Praktični primeri**

**Primer 6.** Na početku, baza podataka je imala tri redo log fajla odnosno tri redo log grupe. Kada su dodata još tri fajla naredbom ALTER DATABASE ADD LOGFILE ... ukupan broj redo log grupa povećan je na šest. Prve tri grupe bile su neaktivne (inactive), četvrta grupa je bila trenutno aktivna (current), a peta i šesta grupa nisu korišćene (unused). Nakon toga, izvršena je naredba ALTER SYSTEM SWITCH LOGFILE. Ova naredba omogućava prelazak sa trenutno aktivnog redo log fajla na sledeći dostupni redo log fajl. U ovom slučaju, to je preusmerilo aktivnost sa četvrte na petu redo log grupu. Kao rezultat, četvrta grupa postala je neaktivna, a peta grupa postala je trenutno aktivna. Ova naredba je ključna za rotaciju redo log fajlova i kontinuirano snimanje promena u bazi podataka. Omogućava bazi podataka da nastavi sa radom bez prekida, čak i kada se jedan set redo log fajlova ispuni, osiguravajući tako kontinuitet i integritet sistema.







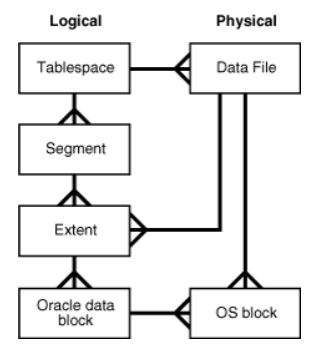


## Logičke strukture skladištenja

Ovo poglavlje opisuje prirodu i odnose među logičkim skladišnim strukturama. Ove strukture su kreirane i prepoznate od strane Oracle baze podataka i nisu poznate operativnom sistemu. Oracle baza podataka dodeljuje logički prostor za sve podatke u bazi podataka.

Logičke jedinice alokacije prostora u bazi podataka su blokovi podataka, segmenti, ekstenzije i tablespace-ovi. Na fizičkom nivou, podaci se skladište u datotekama podataka na disku. Podaci u datotekama podataka se skladište u blokovima operativnog sistema.

Sledeća slika (3.7) je dijagram entiteta i odnosa za fizičko i logičko skladište.



**Slika 3.7** Logičko i fizičko skladištenje

Segment sadrži jedan ili više ekstenzija, pri čemu svaka ekstenzija sadrži više blokova podataka.

Sledeća slika (3.8) prikazuje odnose među blokovima podataka, ekstenzijama i segmentima unutar tablespace-a. U ovom primeru, segment ima dve ekstenzije koje su smeštene u različitim datotekama podataka.



**Slika 3.8** Segmenti, ekstenzije i blokovi podataka unutar tablespace-a

Od najnižeg nivoa granularnosti do najvišeg, Oracle baza podataka čuva podatke.

***Blok podataka***je najmanja logička jedinica skladištenja podataka u Oracle bazi podataka. Jedan logički blok podataka odgovara određenom broju bajtova fizičkog prostora na disku, na primer, 2 KB. Blokovi podataka su najmanje jedinice skladištenja koje Oracle baza podataka može koristiti ili dodeliti.

***Ekstenzija*** je skup logičkih susednih blokova podataka dodeljenih za skladištenje određenog tipa informacija. U prethodnom grafičkom prikazu, ekstenzija od 24 KB ima 12 blokova podataka, dok ekstenzija od 72 KB ima 36 blokova podataka.

***Segment*** je skup ekstenzija dodeljen za određeni objekat baze podataka, kao što je tabela. Na primer, podaci za tabelu zaposlenih se skladište u svom segmentu podataka, dok se svaki indeks za zaposlene skladišti u svom segmentu indeksa. Svaki objekat baze podataka koji koristi skladištenje sastoji se od jednog segmenta.

***Tablespace*** je jedinica skladištenja baze podataka koja sadrži jedan ili više segmenata. Svaki segment pripada samo jednom tablespace-u. Stoga, sve ekstenzije za segment su smeštene u istom tablespace-u. Unutar tablespace-a, segment može uključivati ekstenzije iz više datoteka podataka, kao što je prikazano u prethodnom grafičkom prikazu. Na primer, jedna ekstenzija za segment može biti smeštena u users01.dbf, dok je druga smeštena u users02.dbf. Jedna ekstenzija nikada ne može obuhvatiti više datoteka podataka.

### Upravljanje logičkim prostorom

Oracle baza podataka mora koristiti upravljanje logičkim prostorom kako bi pratila i dodeljivala ekstenzije u tablespace-u. Kada objekat baze podataka zahteva ekstenziju, baza podataka mora imati metodu pronalaženja i pružanja iste. Slično tome, kada objekt više ne zahteva ekstenziju, baza podataka mora imati metodu oslobađanja slobodne ekstenzije. Oracle baza podataka upravlja prostorom unutar tablespace-a na osnovu tipa koji je kreiran. Moguće je kreirati jedan od sledećih tipova tablespace-a:

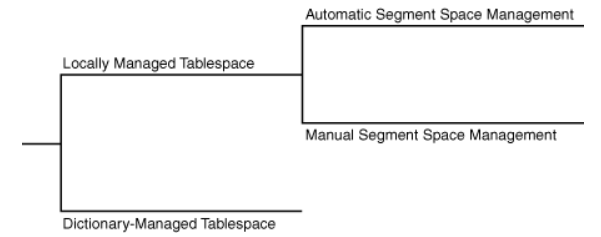
* **Lokalno upravljani tablespace-ovi (podrazumevano)**

Baza podataka koristi bitmapu unutar samih tablespace-ova kako bi upravljala ekstenzijama. Tako lokalno upravljani tablespace-ovi imaju deo prostora rezervisan za bitmapu. Unutar tablespace-a, baza podataka može upravljati segmentima sa automatskim upravljanjem prostorom segmenta (ASSM) ili ručnim upravljanjem prostorom segmenta (MSSM).

* **Tablespace-ovi upravljani rečnikom (Dictionary-managed tablespaces)**

Baza podataka koristi rečnik podataka za upravljanje ekstenzijama.

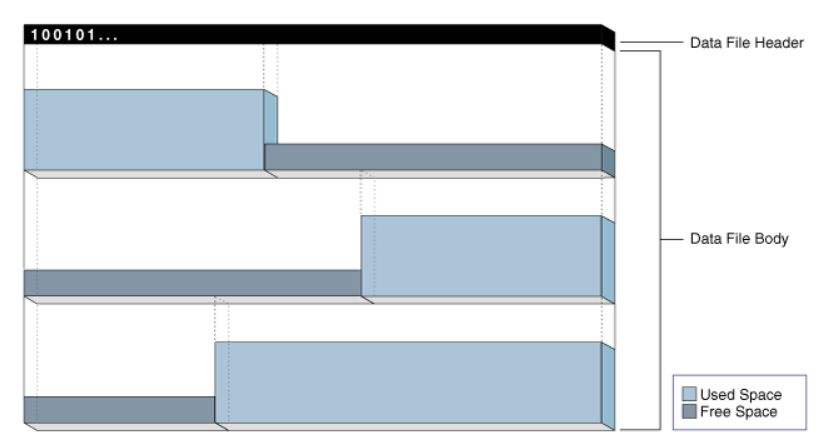
Na slici 3.9 su prikazane alternative za upravljanje logičkim prostorom u tablespace-u.



**Slika 3.9** Upravljanje logičkim prostorom

***Lokalno upravljani tablepsace*** održava bitmapu u zaglavlju datoteke podataka kako bi pratila slobodan i zauzet prostor u telu datoteke podataka. Svaki bit odgovara grupi blokova. Kada se prostor dodeljuje ili oslobađa, Oracle baza podataka menja vrednosti bitmape kako bi odražavala novi status blokova.

Sledeći grafički prikaz (slika 3.10) je konceptualna reprezentacija upravljanja bitmapom. 1 u zaglavlju označava zauzet prostor, dok 0 označava slobodan prostor.



**Slika 3.10** Upravljanje prostorom putem bitmape

Lokalno upravljani tablespace ima sledeće prednosti:

* Izbegava korišćenje rečnika podataka za upravljanje ekstenzijama

Rekurzivne operacije mogu se desiti u tablespace-ovima upravljanim rečnikom ako korišćenje ili oslobađanje prostora u ekstenziji rezultira drugom operacijom koja koristi ili oslobađa prostor u tabeli rečnika podataka ili undo segmentu.

* Automatski prati slobodan prostor koji je uzastopan

Na ovaj način, baza podataka eliminiše potrebu za spajanjem slobodnih ekstenzija.

* Automatski određuje veličinu lokalno upravljanih ekstenzija

Alternativno, sve ekstenzije mogu imati istu veličinu u lokalno upravljanom tablespace-u i prebrisati opcije skladištenja objekta.

Upravljanje prostorom segmenta je atribut koji se nasleđuje od tablespace-a koji sadrži segment. Unutar lokalno upravljanih tablespace-ova, baza podataka može automatski ili ručno upravljati segmentima. Na primer, segmenti u tabelarnom prostoru users mogu biti automatski upravljani dok su segmenti u tablespace-u tools ručno upravljani.

***Metod automatskog upravljanja prostorom segmenta (ASSM)*** koristi bitmapu za upravljanje prostorom u tablespace-u. Bitmapa pruža sledeće prednosti:

* Jednostavnija administracija

ASSM izbegava potrebu za ručnim određivanjem ispravnih postavki za mnoge parametre skladištenja. Samo jedan ključni SQL parametar kontroliše dodelu prostora: PCTFREE. Ovaj parametar određuje procenat prostora koji treba rezervisati u bloku za buduće ažuriranje.

* Povećana paralelnost

Više transakcija može pretraživati odvojene liste slobodnih blokova podataka, čime se smanjuje sukobljenost i čekanja. Za mnoga standardna radna opterećenja, performanse aplikacije sa ASSM su bolje nego performanse dobro podešene aplikacije koja koristi MSSM.

* Dinamična povezanost prostora sa instancama u okruženju Oracle Real Application Clusters (Oracle RAC)

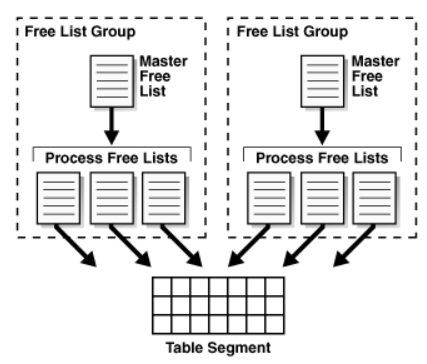
ASSM je efikasniji i podrazumevan je za trajne, lokalno upravljanog tablespace-a.

***Metod ručnog upravljanja prostorom segmenta (MSSM)*** koristi povezanu listu nazvanu slobodna lista kako bi upravljao slobodnim prostorom u segmentu. Za bazu podataka koja ima slobodan prostor, slobodna lista prati blokove ispod visoke vodene oznake (HWM), što je linija razdvajanja između segmentnog prostora koji je korišćen i još uvek nekorišćen. Kako se blokovi koriste, baza podataka dodaje blokove na slobodnu listu ili ih uklanja prema potrebi.

Pored PCTFREE, MSSM zahteva da upravljate dodelom prostora pomoću SQL parametara kao što su PCTUSED, FREELISTS i FREELIST GROUPS. PCTUSED postavlja procenat slobodnog prostora koji mora postojati u trenutno korišćenom bloku kako bi ga baza podataka stavila na slobodnu listu. Na primer, ako se postavi PCTUSED na 40 u CREATE TABLE naredbi, tada nije moguće ubaciti redove u blok u segmentu dok se ne koristi manje od 40% prostora bloka.

Na primer, pretpostavimo da se ubacuje red u tabelu. Baza podataka proverava slobodnu listu tabele za prvi dostupni blok. Ako red ne stane u blok i ako je korišćeni prostor u bloku veći ili jednak PCTUSED, tada baza podataka uklanja blok sa liste i traži drugi blok. Ako se brišu redovi iz bloka, tada baza podataka proverava da li je korišćeni prostor u bloku sada manji od PCTUSED. Ako jeste, tada baza podataka stavlja blok na početak slobodne liste.

Objekat može imati više slobodnih listi. Na ovaj način, više sesija koje izvršavaju DML na tabeli mogu koristiti različite liste, što može smanjiti sukobljenost. Svaka sesija baze podataka koristi samo jednu slobodnu listu tokom trajanja svoje sesije. Kao što je prikazano na slici 3.11, moguće je kreirati objekat sa jednom ili više grupa slobodnih listi, koje su kolekcije slobodnih listi. Svaka grupa ima glavnu slobodnu listu koja upravlja pojedinačnom slobodnom listom procesa u grupi. Prostorni overhead za slobodne liste, posebno za grupe slobodnih listi, može biti značajan.



**Slika 3.11** Grupe slobodnih lista

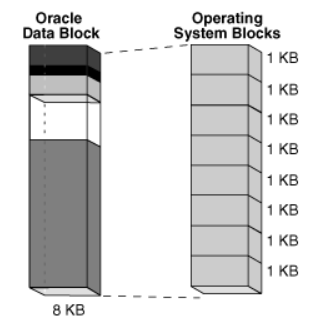
Upravljanje prostorom segmenta ručno može biti kompleksno. Potrebno je podešavati PCTFREE i PCTUSED kako bi smanjili migraciju redova i izbegli rasipanje prostora. Na primer, ako su svi korišćeni blokovi u segmentu pola puni, a ako je PCTUSED 40, tada baza podataka ne dozvoljava ubacivanje u bilo koji od tih blokova. Zbog zahtevnosti finog podešavanja parametara dodele prostora, Oracle preporučuje korišćenje ASSM-a. U ASSM-u, PCTFREE određuje da li novi red može biti ubačen u blok, ali ne koristi slobodne liste i ignoriše PCTUSED.

***Tablespace-ovi upravljani rečnikom*** koriste rečnik podataka za upravljanje svojim ekstenzijama. Oracle baza podataka ažurira tabele u rečniku podataka svaki put kada se ekstenzija dodeli ili oslobodi za ponovnu upotrebu. Na primer, kada je tabeli potrebna ekstenzija, baza podataka pretražuje tabele u rečniku podataka i traži slobodne ekstenzije. Ako baza podataka pronađe prostor, tada vrši modifikaciju jedne tabele u rečniku podataka i ubacuje red u drugu tabelu. Na ovaj način, baza podataka upravlja prostorom modifikujući i premestajući podatke. SQL koji baza podataka izvršava u pozadini kako bi dobila prostor za objekte baze podataka naziva se rekurzivni SQL. Često korišćenje rekurzivnog SQL-a može imati negativan uticaj na performanse jer se ažuriranja u rečniku podataka moraju serijski izvršavati. Lokalno upravljani tablespace-ovi, koji su podrazumevani, izbegavaju ovaj problem performansi.

### Blokovi podataka

Na fizičkom nivou, podaci baze podataka se skladište u disk datotekama koje se sastoje od blokova operativnog sistema. Blok operativnog sistema je minimalna jedinica podataka koju operativni sistem može čitati ili pisati. Nasuprot tome, Oracle blok je logička skladišna struktura čija veličina i struktura nisu poznate operativnom sistemu.

Sledeća slika (3.12) pokazuje da blokovi operativnog sistema mogu imati različite veličine u odnosu na blokove podataka. Baza podataka zahteva podatke u višestrukim brojevima blokova podataka, ne blokova operativnog sistema.



**Slika 3.12** Blokovi podataka i blokovi operativnog sistema

Kada baza podataka zahteva blok podataka, operativni sistem prevodi ovu operaciju u zahtev za podacima u trajnom skladištenju. Logička razdvojenost blokova podataka od blokova operativnog sistema ima sledeće implikacije:

* Aplikacije ne moraju da određuju fizičke adrese podataka na disku.
* Podaci baze podataka mogu biti prugasti ili preslikani na više fizičkih diskova.

Svaka baza podataka ima veličinu bloka podataka. Parametar inicijalizacije DB\_BLOCK\_SIZE postavlja veličinu bloka podataka za bazu podataka prilikom njenog kreiranja. Veličina se postavlja za tablespace SYSTEM i SYSAUX i podrazumevana je za sve ostale tablespace-ove. Veličina bloka podataka baze podataka ne može biti promenjena osim ako se ne kreira baza podataka ponovo. Ako DB\_BLOCK\_SIZE nije postavljen, tada je podrazumevana veličina bloka podataka specifična za operativni sistem. Standardna veličina bloka podataka za bazu podataka je 4 KB ili 8 KB. Ako se veličina razlikuje za blokove podataka i blokove operativnog sistema, tada veličina bloka podataka mora biti višestruka veličina bloka operativnog sistema.

Moguće je kreirati individualne tablespace-ove čija se veličina bloka razlikuje od podešavanja DB\_BLOCK\_SIZE. Nestandardna veličina bloka može biti korisna prilikom prenosa transportabilnih tablespace-ova na drugu platformu.

Svaki blok podataka ima format ili internu strukturu koja omogućava bazi podataka da prati podatke i slobodan prostor u bloku. Ovaj format je sličan bez obzira da li blok podataka sadrži podatke tabele, indeksa ili tabele klastera. Sledeća slika (3.13) prikazuje format nekompresovanog bloka podatka.



**Slika 3.13** Format nekompresovanog bloka podataka

Oracle baza podataka koristi blokove za upravljanje samim blokovima. Blokovi nisu dostupni za skladištenje korisničkih podataka.

Kao što je prikazano na slici 3.13, blokovi uključuju sledeće delove:

* **Zaglavlje bloka**

Ovaj deo sadrži opšte informacije o bloku, uključujući adresu na disku i tip segmenta. Za blokove koji su upravljani transakcijama, zaglavlje bloka sadrži informacije o aktivnim i istorijskim transakcijama. Transakcioni unos je potreban za svaku transakciju koja ažurira blok. Oracle baza podataka rezerviše prostor u zaglavlju bloka za transakcione unose. U blokovima podataka alociranim segmentima koji podržavaju transakcione promene, slobodni prostor takođe može sadržati transakcione unose kada je prostor u zaglavlju iscrpljen. Prostor potreban za transakcione unose zavisi od operativnog sistema. Međutim, transakcioni unosi na većini operativnih sistema zahtevaju otprilike 23 bajta.

* **Direktorijum tabele**

Za tabelu organizovanu kao stog, ovaj direktorijum sadrži metapodatke o tabelama čije su redovi smešteni u ovom bloku. U klasteru tabela, više tabela može skladištiti redove u istom bloku.

* **Direktorijum redova**

Za tabelu organizovanu kao heap, ovaj direktorijum opisuje lokaciju redova u delu bloka podataka. Baza podataka može postaviti red bilo gde na dnu bloka. Adresa reda je zabeležena u jednom od slotova vektora direktorijuma redova. Rowid pokazuje na određenu datoteku, blok i redni broj reda.

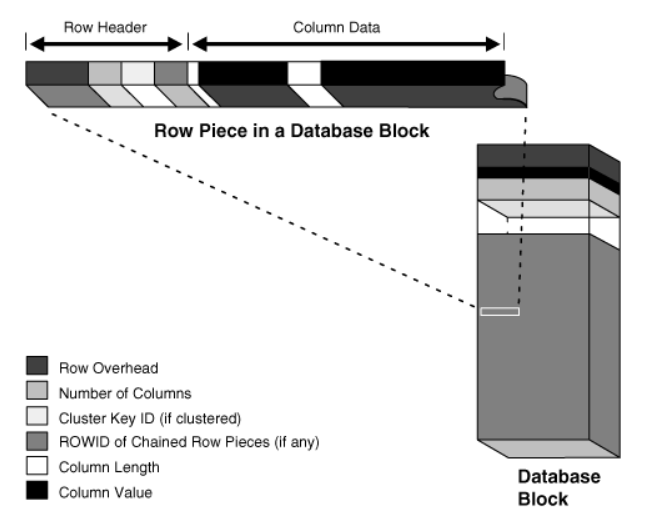
Na primer, u rowid AAAPecAAFAAAABSAAA, poslednje AAA predstavlja redni broj reda. Redni broj reda je indeks u vektoru direktorijuma redova. Unos direktorijuma redova sadrži pokazivač na lokaciju reda u bloku podataka. Ako baza podataka pomeri red unutar bloka, baza podataka ažurira unos direktorijuma redova kako bi promenila pokazivač. Rowid ostaje konstantan.

Nakon što baza podataka alocira prostor u direktorijumu redova, baza podataka ne vraća ovaj prostor nakon brisanja redova. Dakle, blok koji je trenutno prazan, ali koji je nekad imao do 50 redova i dalje ima alociranih 100 bajtova za direktorijum redova. Baza podataka ponovno koristi ovaj prostor samo kada sesija unese nove redove u blok.

Neke delove blokova su fiksne veličine, ali ukupna veličina je promenljiva. Prosečno, blokovi sadrže ukupno 84 do 107 bajtova blokiranog overheada.

Deo podataka o redovima u bloku sadrži stvarne podatke, poput redova tabele ili unosa ključeva indeksa. Baš kao što svaki blok podataka ima interni format, svaki red ima format reda koji omogućava bazi podataka da prati podatke u redu. Oracle baza podataka čuva redove kao zapise promenljive dužine. Red je sadržan u jednom ili više delova. Svaki deo se naziva **redni segment**. Svaki redni segment ima zaglavlje reda i podatke o koloni.

Sledeća slika (3.14) prikazuje format reda.



**Slika 3.14** Format rednog segmenta

Oracle baza podataka koristi zaglavlje reda za upravljanje rednim segmentom koji je smešten u bloku. Zaglavlje reda sadrži informacije kao što su:

* Kolone u rednom segmentu
* Segmenti reda smešteni u drugim blokovima podataka

Ako se ceo red može ubaciti u jedan blok podataka, tada Oracle baza podataka čuva red kao jedan redni segment. Međutim, ako svi podaci o redu ne mogu biti ubačeni u jedan blok ili ažuriranje uzrokuje da postojeći red preraste svoj blok, tada baza podataka čuva red u više rednih segmenata. Blok podataka obično sadrži samo jedan redni segment po redu.

* Klaster ključevi za klaster tabele

Red koji je u potpunosti sadržan u jednom bloku ima najmanje 3 bajta zaglavlja reda.

Nakon zaglavlja reda, deo podataka o koloni čuva stvarne podatke u redu. Redni segment obično čuva kolone u redosledu navedenom u CREATE TABLE naredbi, ali ovaj redosled nije garantovan.

Na primer, kolone tipa LONG se kreiraju poslednje. Kao što je prikazano na slici 3.14, za svaku kolonu u rednom segmentu, Oracle baza podataka čuva dužinu kolone i podatke odvojeno. Potrebni prostor zavisi od tipa podataka. Ako je tip podataka kolone promenljive dužine, tada prostor potreban za čuvanje vrednosti može rasti i smanjivati se sa ažuriranjima podataka. Svaki red ima slot u direktorijumu reda zaglavlja bloka podataka. Slot pokazuje na početak reda.

Oracle baza podataka koristi rowid da jedinstveno identifikuje reda. Interno, rowid je struktura koja sadrži informacije koje su bazi podataka potrebne da pristupi redu. Rowid se ne čuva fizički u bazi podataka, već se izvodi iz datoteke i bloka u kojem su podaci smešteni. Prošireni rowid uključuje broj objekta podataka. Ovaj tip rowid-a koristi kodiranje baze 64 fizičke adrese za svaki red. Karakteri kodiranja su A-Z, a-z, 0-9, + i /.

**Primer 7:** *Pseudokolona ROWID*

Sledeći primer koristi upit na pseudokolonu ROWID kako bi prikazao prošireni rowid reda u tabeli employees za zaposlenog 100:

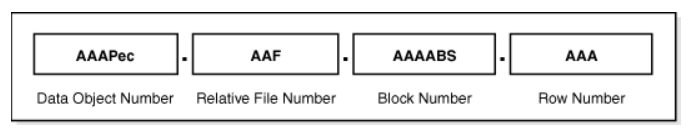
SQL> SELECT ROWID FROM employees WHERE employee\_id = 100;

ROWID

------------------

AAAPecAAFAAAABSAAA

Sledeća slika (3.15) ilustruje format proširenog rowid-a.



**Slika 3.15** Format proširenog rowid-a

Prošireni rowid se prikazuje u formatu sa četiri dela, OOOOOOFFFBBBBBBRRR, pri čemu je format podeljen na sledeće komponente:

* OOOOOO

Broj objekta podataka identifikuje segment (objekat podataka AAAPec u primeru upita). Broj objekta podataka je dodeljen svakom segmentu baze podataka. Objekti šeme u istom segmentu, kao što je klaster tabele, imaju isti broj objekta podataka.

* FFF

Broj datoteke u odnosu na tablespace identifikuje datoteku koja sadrži red (datoteka AAF u primeru upita).

* BBBBBB

Broj bloka podataka identifikuje blok koji sadrži red (blok AAAABS u primeru upita). Brojevi blokova su relativni u odnosu na svoju datoteku podataka, ne na svoj tablespace. Dakle, dva reda sa identičnim brojevima blokova mogu biti smešteni u različitim datotekama podataka istog tablespace-a.

* RRR

Broj reda identifikuje red u bloku (red AAA u primeru upita).

Nakon što se rowid dodeli rednom segmentu, rowid može promeniti u posebnim okolnostima. Na primer, ako je omogućeno kretanje redova, tada rowid može promeniti zbog ažuriranja ključeva particije, operacija Flashback Table, operacija smanjenja tabele, itd. Ako je kretanje redova onemogućeno, tada se rowid može promeniti ako se red izveze i uveze korišćenjem Oracle baza podataka alatki.

Baza podataka može koristiti **kompresiju tabele** da eliminiše duplirane vrednosti u bloku podataka. Ova sekcija opisuje format blokova podataka koji koriste kompresiju. Format bloka podataka koji koristi osnovnu tabelu i naprednu kompresiju redova je suštinski isti kao i nekompresovan blok. Razlika je u tome što se na početku bloka nalazi tabela simbola koja čuva duplirane vrednosti za redove i kolone. Baza podataka zamenjuje pojavljivanja ovih vrednosti skraćenim referencama ka tabeli simbola.

**Primer 8:** *Format kompresovanih blokova podataka*

Pretpostavimo da su sledeći redovi smešteni u bloku podataka za tabelu sales sa sedam kolona:

2190,13770,25-NOV-00,S,9999,23,161

2225,15720,28-NOV-00,S,9999,25,1450

34005,120760,29-NOV-00,P,9999,44,2376

9425,4750,29-NOV-00,I,9999,11,979

1675,46750,29-NOV-00,S,9999,19,1121

Kada se primeni osnovna tabela ili napredna kompresija redova na ovu tabelu, baza podataka zamenjuje duplirane vrednosti referencama ka simbolima. Sledeći konceptualni prikaz kompresije pokazuje simbol \* koji zamenjuje 29-NOV-00 i % koji zamenjuje 9999:

2190,13770,25-NOV-00,S,%,23,161

2225,15720,28-NOV-00,S,%,25,1450

34005,120760,\*,P,%,44,2376

9425,4750,\*,I,%,11,979

1675,46750,\*,S,%,19,1121

Sledeća tabela konceptualno predstavlja tabelu simbola koja mapira simbole na vrednosti.

|  |  |  |  |
| --- | --- | --- | --- |
| Symbol | Value | Column | Rows |
| \* | 29-NOV-00 | 3 | 958-960 |
| 5 | 9999 | 5 | 956-960 |

**Index blok** je poseban tip bloka podataka koji upravlja prostorom na drugačiji način u odnosu na blokove tabela. Oracle baza podataka koristi index blokove za upravljanje logičkim prostorom skladištenja u indeksu. Indeks sadrži root blok, branch blokove i leaf blokove. Tipovi blokova su definisani na sledeći način:

* **Root blok**

Ovaj blok identifikuje ulaznu tačku u indeks.

* **Branch blokovi**

Baza podataka prolazi kroz branch blokove prilikom traženja ključa indeksa.

* **Leaf blokovi**

Ovi blokovi sadrže vrednosti ključeva indeksa i rowid-ove koji pokazuju na povezane redove. Leaf blokovi čuvaju vrednosti ključeva u sortiranom redosledu tako da baza podataka može efikasno tražiti sve redove u opsegu vrednosti ključeva.

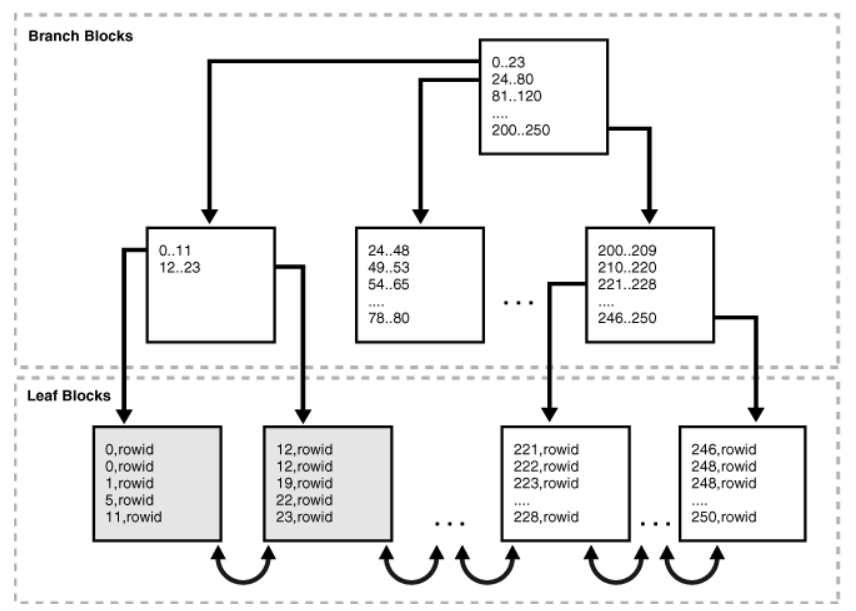
Spajanje indeksnog bloka kompresuje postojeće podatke indeksa na licu mesta iako reorganizacija oslobodi blokove, ostavlja slobodne blokove u strukturi indeksa. Dakle, spajanje ne oslobađa blokove indeksa za druge svrhe niti uzrokuje realokaciju blokova indeksa. Oracle baza podataka ne kompresuje indeks automatski: mora se izvršiti ALTER INDEX naredba sa opcijama REBUILD ili COALESCE.

Slika 3.16 prikazuje indeks kolone employees.department\_id pre nego što je indeks spojen. Prva tri list bloka su samo delimično popunjena, što je naznačeno sivim linijama.



**Slika 3.16** Indeks pre spajanja

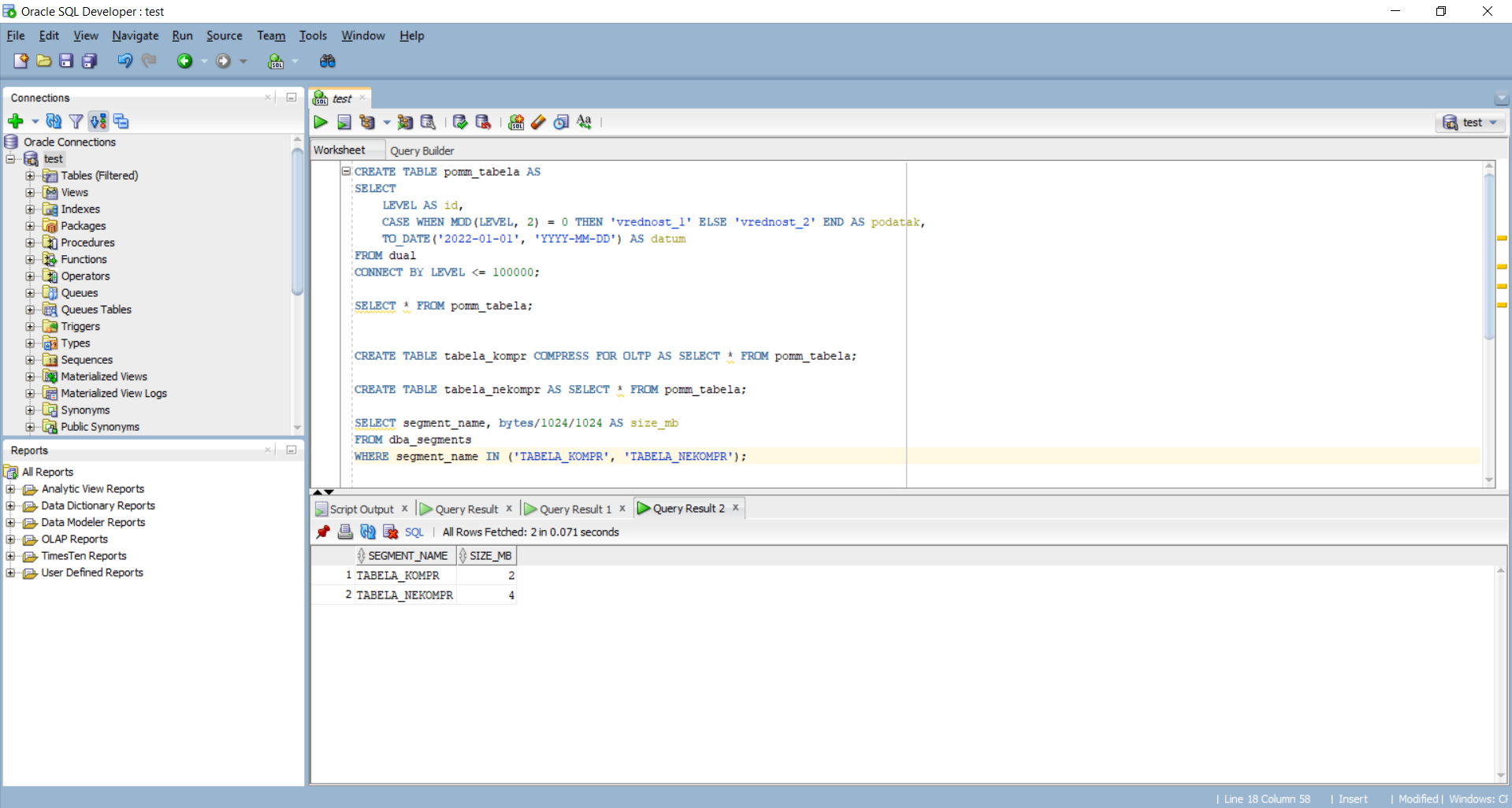
Sledeća slika (3.17) prikazuje indeks sa slike 3.16 nakon što je indeks spojen. Prva dva leaf bloka sada su puna, što je naznačeno sivim linijama, dok je treći leaf blok oslobođen.



**Slika 3.17** Indeks nakon spajanja

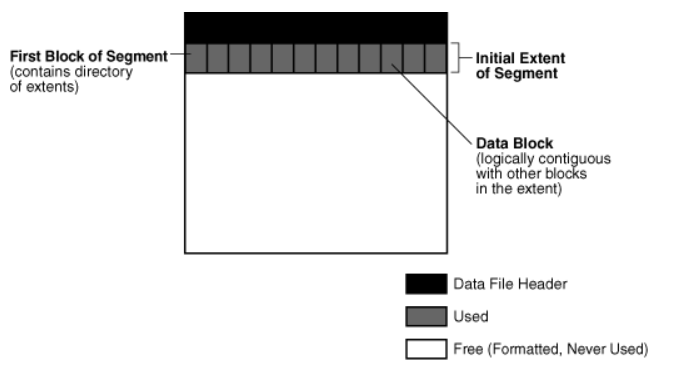
**Praktični primeri**

**Primer 9.** Sledeći primer simulirana situaciju u kojoj su kreirane kompresovana i nekompresovana verzija istih podataka kako bi se uporedila njihova veličina na disku. Korišćena je fiktivna tabela sa generisanim podacima kako bi se demonstrirao efekat kompresije. Nakon kreiranja obe tabele, njihova veličina je prikazana radi poređenja.



### Ekstenzije

Podrazumevano, baza podataka dodeljuje početnu ekstenziju za segment podataka kada se segment kreira. Ekstenzija je uvek sadržana u jednoj datoteci podataka. Iako podaci nisu dodati u segment, blokovi podataka u početnoj ekstenziji su rezervisani isključivo za taj segment. Prvi blok podataka svakog segmenta sadrži direktorijum ekstenzija u segmentu. Slika 3.17 prikazuje početnu ekstenziju u segmentu u datoteci podataka koja prethodno nije sadržavala podatke.

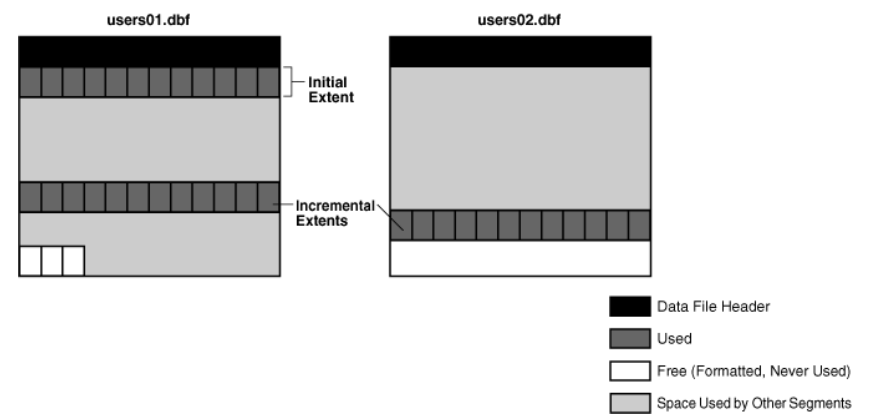


**Slika 3.18** Početna ekstenzija segmenta

Ako početna ekstenzija postane puna i ako je potrebno više prostora, tada baza podataka automatski dodeljuje inkrementalnu ekstenziju za ovaj segment. Inkrementalna ekstenzija je naredna ekstenzija kreirana za segment.

Algoritam dodele zavisi od toga da li je tablespace upravljan lokalno ili upravljan rečnikom. U slučaju lokalnog upravljanja, baza podataka pretražuje bit mapu datoteke podataka za susedne slobodne blokove. Ako datoteka podataka nema dovoljno prostora, tada baza podataka traži u drugoj datoteci podataka. Ekstenzije za segment su uvek u istom tablespace-u ali mogu biti u različitim datotekama podataka. Slika 3.19 pokazuje da baza podataka može dodeliti ekstenzije za segment u bilo kojoj datoteci podataka u tablespace-u.

Na primer, segment može dodeliti početnu ekstenziju u users01.dbf, dodeliti prvu inkrementalnu ekstenziju u users02.dbf, a zatim dodeliti sledeću ekstenziju u users01.dbf.



**Slika 3.19** Inkrementalna ekstenzija segmenta

Blokovi novo dodeljene ekstenzije, iako su bili slobodni, možda neće biti prazni. U ASSM-u, Oracle baza podataka formatira blokove novo dodeljene ekstenzije kada počne da koristi ekstenziju, ali samo po potrebi.

### Segmenti

**Korisnički segmenti**

Jedan podatak segmenta u bazi podataka skladišti podatke za jedan korisnički objekat. Postoje različite vrste segmenata. Primeri korisničkih segmenata uključuju:

* Tabela, particija tabele ili klaster tabele
* LOB [[3]](#footnote-3)ili particija LOB-a
* Indeks ili particija indeksa

Svaki ne-particionisani objekat i particija objekta se čuva u svom segmentu. Na primer, ako indeks ima pet particija, tada pet segmenata sadrži podatke indeksa.

Podrazumevano, baza podataka koristi odloženo kreiranje segmenata kako bi ažurirala samo metapodatke baze podataka prilikom kreiranja tabela, indeksa i particija. Kada korisnik doda prvi red u tabelu ili particiju, baza podataka kreira segmente za tu tabelu ili particiju, njene LOB kolone i njene indekse. Odloženo kreiranje segmenata izbegava nepotrebnu upotrebu resursa baze podataka. Na primer, instalacija aplikacije može kreirati hiljade objekata, što zauzima značajan prostor na disku. Mnogi od tih objekata možda nikada neće biti korišćeni.

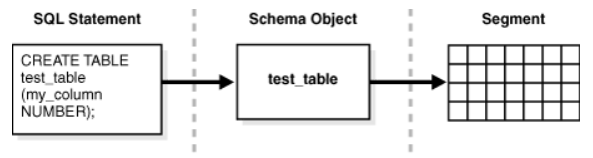
Paket DBMS\_SPACE\_ADMIN upravlja segmentima za prazne objekte. Moguće je koristiti ovaj PL/SQL paket kako bi:

* Ručno materijalizovali segmente za prazne tabele ili particije koje nemaju kreirane segmente
* Uklonili segmente iz praznih tabela ili particija koji trenutno imaju prazan dodeljen segment

kako bi najbolje ilustrovali odnos između kreiranja objekata i kreiranja segmenata, pretpostavimo da je odloženo kreiranje segmenata onemogućeno. Kreiramo tabelu na sledeći način:

CREATE TABLE test\_table (my\_column NUMBER);

Kao što je prikazano na slici 3.20, baza podataka kreira jedan segment za tabelu.

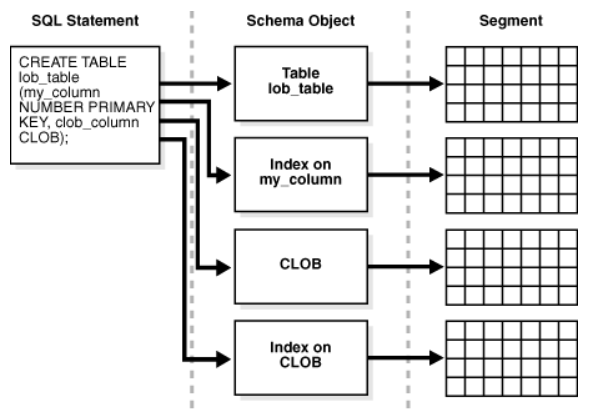


**Slika 3.20** Kreiranje korisničkog segmenta

Kada kreiramo tabelu sa primarnim ključem ili jedinstvenim ključem, Oracle baza podataka automatski kreira indeks za ovaj ključ. Ponovo pretpostavimo da je odloženo kreiranje segmenata onemogućeno. Kreiramo tabelu na sledeći način:

CREATE TABLE lob\_table (my\_column NUMBER PRIMARY KEY, clob\_column CLOB);

Slika 3.21 pokazuje da su podaci za lob\_table smešteni u jednom segmentu, dok je implicitno kreirani indeks smešten u drugom segmentu. Takođe, CLOB podaci se čuvaju u svom segmentu, kao i njihov povezani CLOB indeks. Dakle, CREATE TABLE naredba rezultuje kreiranjem četiri različita segmenta.



**Slika 3.21** Više segmenata

**Privremeni segmenti**

Prilikom obrade upita, Oracle baza podataka često zahteva privremeni radni prostor za privremene faze izvršenja SQL naredbe. Tipične operacije koje mogu zahtevati privremeni segment uključuju sortiranje, heširanje i spajanje bitmapa. Prilikom kreiranja indeksa, Oracle baza podataka takođe smešta indeksne segmente u privremene segmente, a zatim ih konvertuje u trajne segmente kada je indeks kompletiran.

Oracle baza podataka ne kreira privremeni segment ako se operacija može izvršiti u memoriji. Međutim, ako upotreba memorije nije moguća, tada baza podataka automatski dodeljuje privremeni segment na disku.

**Undo segmenti**

Oracle baza podataka čuva zapise akcija transakcija, zajednički poznate kao undo podaci. Oracle baza podataka koristi undo podatke za sledeće:

* Povratak aktivne transakcije
* Obnova završene transakcije
* Obezbeđivanje konzistentnosti čitanja
* Izvršavanje nekih logičkih flashback operacija

Oracle baza podataka čuva undo podatke unutar baze podataka umesto u spoljnim logovima. Undo podaci su smešteni u blokovima koji se ažuriraju baš kao i blokovi podataka, pri čemu promene u ovim blokovima generišu redo zapise. Na ovaj način, Oracle baza podataka može efikasno pristupiti undo podacima bez potrebe za čitanjem spoljnih logova.

Undo podaci za trajne objekte smešteni su u undo tablespace-u. Oracle baza podataka pruža potpuno automatizovan mehanizam, poznat kao automatski režim upravljanja undo-om, za upravljanje undo segmentima i prostorom u undo tablespace-u.

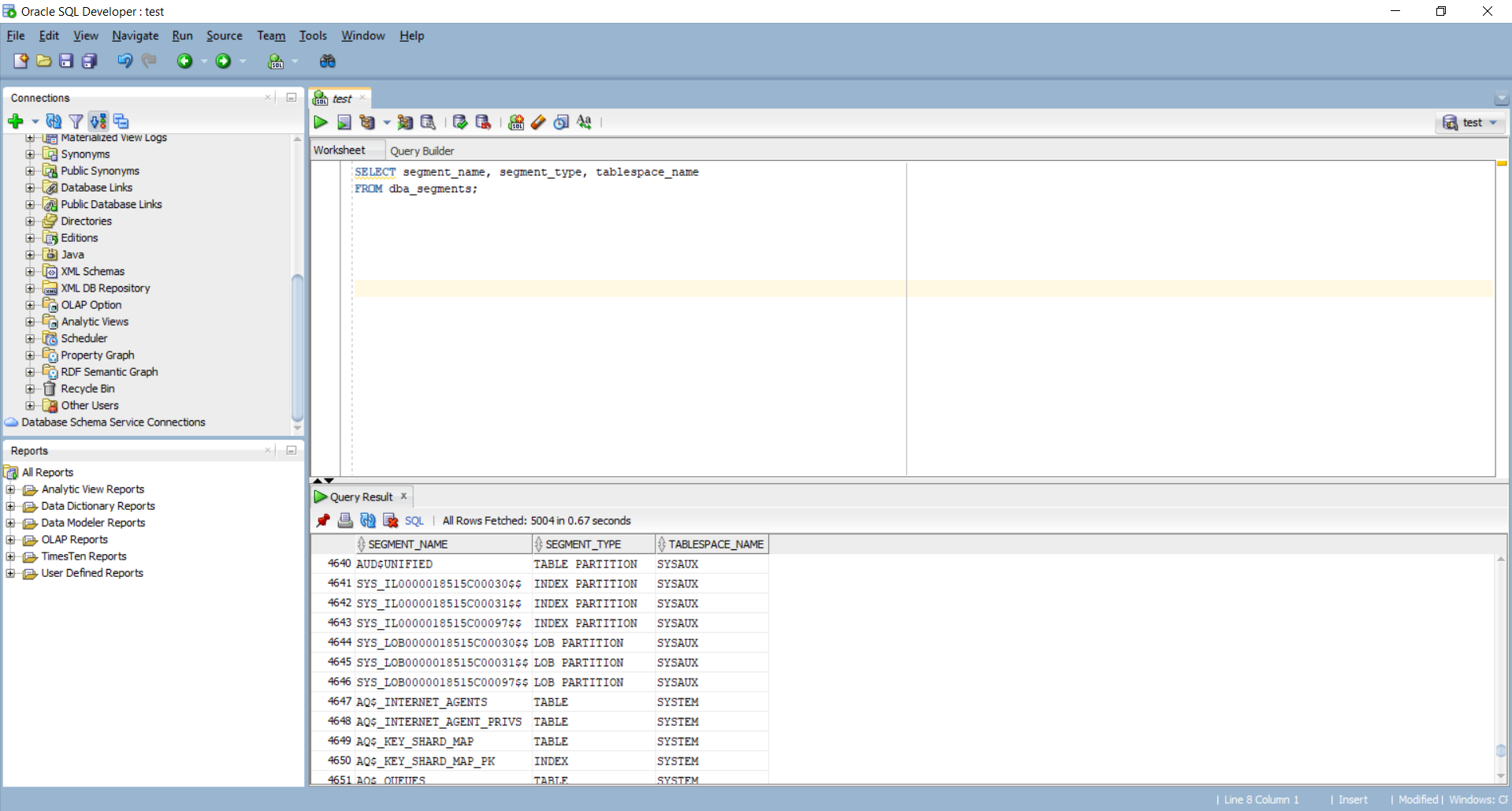
Baza podataka razdvaja undo podatke u dva toka. Privremeni undo tok obuhvata samo undo zapise generisane promenama na privremenim objektima, dok trajni undo tok obuhvata samo undo zapise za trajne objekte. Baza podataka upravlja privremenim i trajnim undo nezavisno. Razdvajanje undo podataka smanjuje potrošnju prostora i poboljšava performanse na sledeći način:

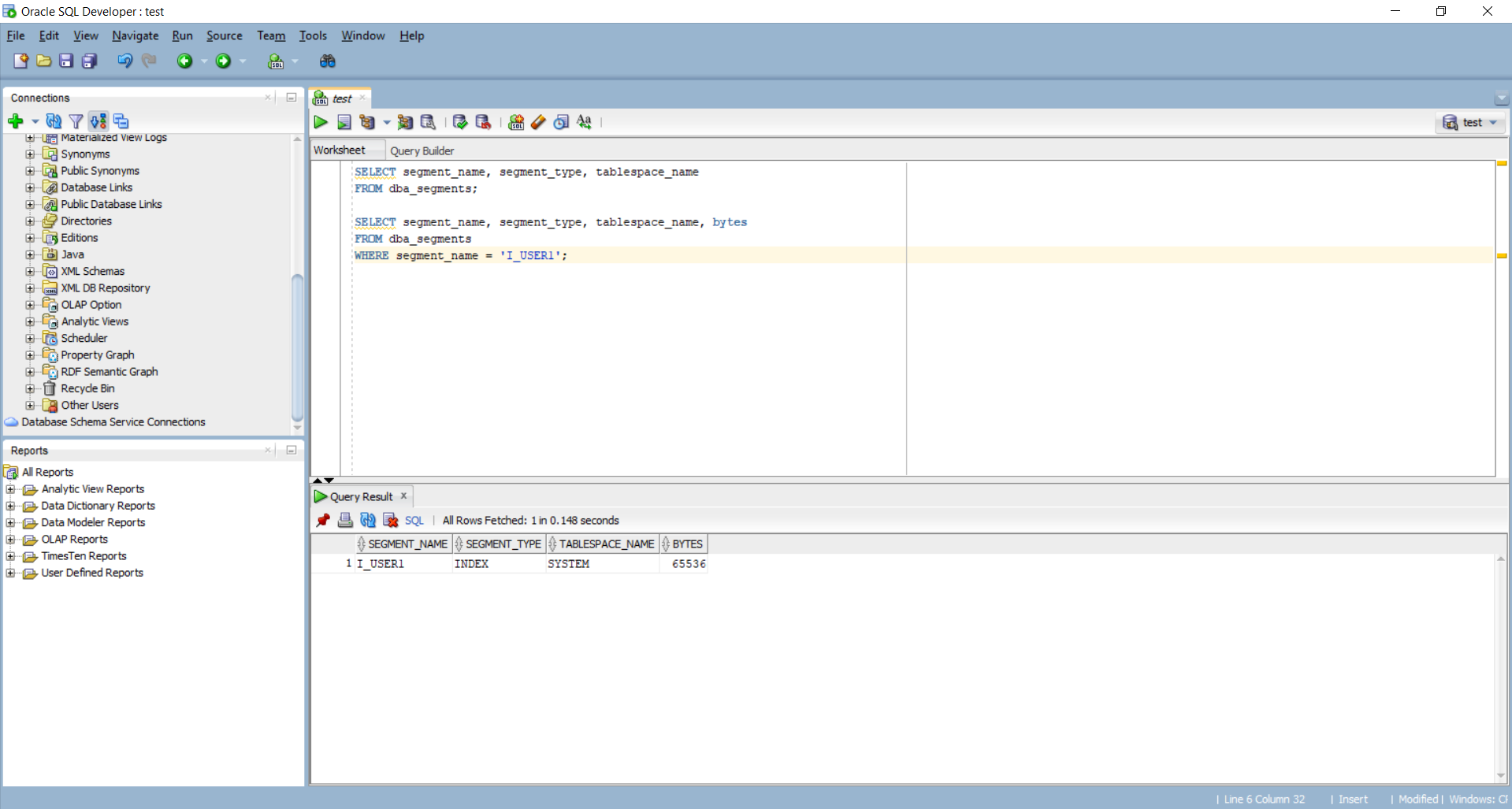
* Omogućava konfigurisanje veličine trajnih i undo tablespace-ova koje najbolje odgovaraju opterećenju za trajne i privremene tabele
* Smanjuje veličinu redo zapisa koji se pišu u online redo log
* Izbegava potrebu za bekapovanjem privremenih undo podataka

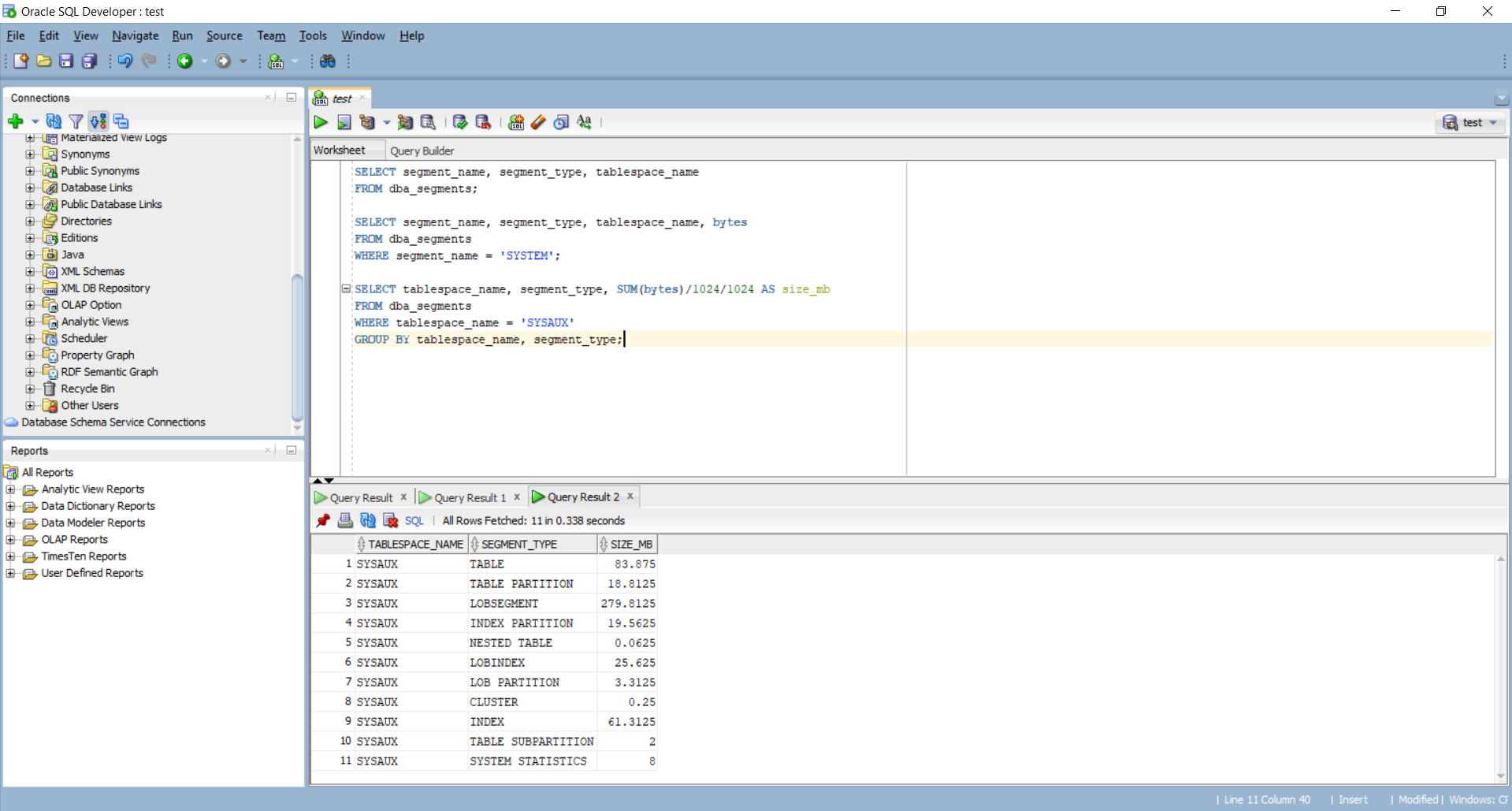
Na Active Data Guard instanci, DML operacije na globalnim privremenim tabelama zahtevaju generisanje undo podataka u privremenim undo segmentima.

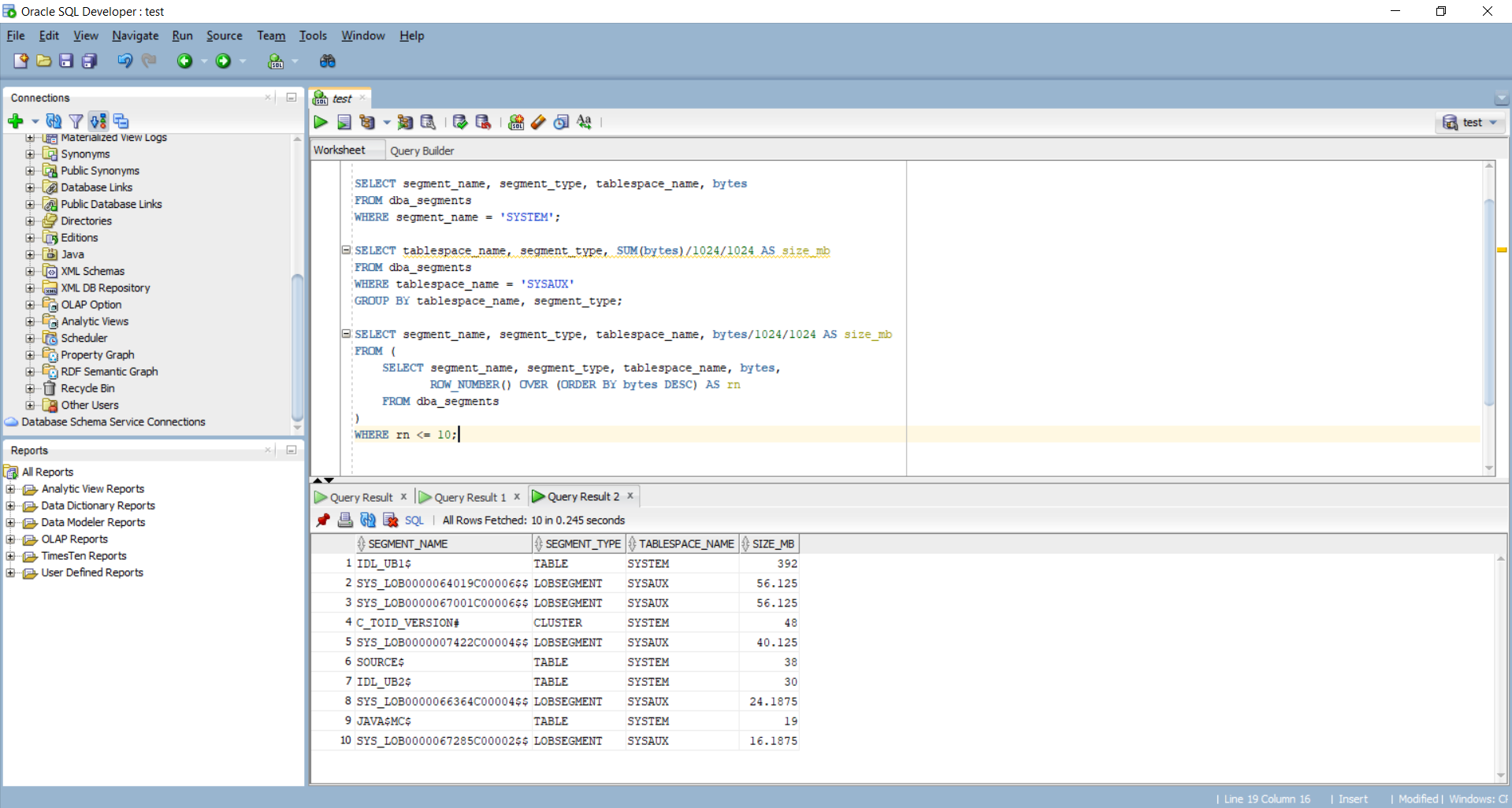
**Praktični primeri**

**Primer 10.** U narednih nekoliko primera prikazano je kako izlistati informacije o segmentima u bazi podataka kao što su prikaz svih segmenata u bazi podataka, prikaz informacija o odredjenom segmentu, prikaz svih segmenata u tablespace-u SYSAUX, prikaz deset najvećih segmenata (po veličini) u bazi podataka.





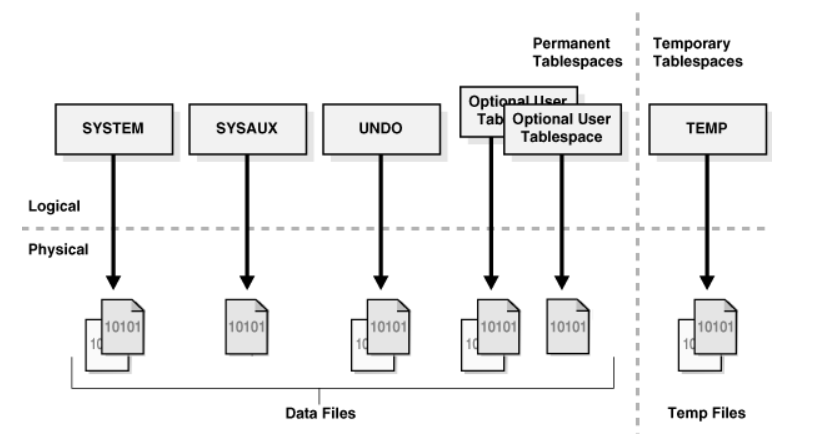




### Tablespace

Tabelspace je logički kontejner za segmente. Segmenti su objekti baze podataka, poput tabela i indeksa, koji troše prostor za skladištenje. Na fizičkom nivou, tabelspace čuva podatke u jednoj ili više datoteka podataka ili privremenih datoteka.

Baza podataka mora imati SYSTEM i SYSAUX tabelspace-ove. Sledeća slika (3.22) prikazuje tabelspace-ove u tipičnoj bazi podataka.



**Slika 3.22** Tablespace-ovi

**Permanentni tablespace-ovi**

Permanentni tablespace grupišu trajne šeme objekata. Segmenti za objekte u tablespace-u fizički se čuvaju u datotekama podataka. Svaki korisnik baze podataka dobija podrazumevani permanentni tablespace. Veoma mala baza podataka možda će zahtevati samo podrazumevane SYSTEM i SYSAUX tablespacove. Međutim, Oracle preporučuje da se kreira bar jedan tablespace za skladištenje korisničkih i aplikacionih podataka. Tabelspace-ovi se mogu koristiti kako bi se postigli sledeći ciljevi:

* Kontrolisanje dodele prostora na disku za podatke baze podataka
* Dodeljivanje kvotu (dozvoljeni prostor ili ograničenje) korisniku baze podataka
* Stavljanje pojedinačnih tablespace-ova online ili offline bez uticaja na dostupnost cele baze podataka
* Vršenje beckup-a i oporavak pojedinačnih tablespace-ova
* Uvez ili izvez aplikacionih podataka koristeći Oracle Data Pump alat
* Kreiranje transportabilnih tablespace-ova koji se mogu kopirati ili premestiti iz jedne baze podataka u drugu, čak i između različitih platformi. Pomeranje podataka putem transporta tablespace-ova može biti mnogo brže nego korišćenje export/import ili unload/load operacija za iste podatke, jer transport tablespace-a uključuje samo kopiranje datoteka podataka i integraciju metapodataka tablespaca. Prilikom transporta tablespaca moguće je premestiti i podatke indeksa.

**SYSTEM tablespace**

SYSTEM tabelspace je neophodan administrativni tabelspace koji je uključen u bazu podataka prilikom njene kreacije. Oracle baza podataka koristi SYSTEM tabelspace za upravljanje bazom podataka. SYSTEM tabelspace uključuje sledeće informacije, sve vlasništvo korisnika SYS:

* Rečnik podataka
* Tabele i prikazi koji sadrže administrativne informacije o bazi podataka
* Kompilovani objekti kao što su okidači, procedure i paketi

SYSTEM tabelspace se upravlja kao i svaki drugi tabelspace, ali zahteva viši nivo privilegija i ograničen je na određene načine. Na primer, ne možemo preimenovati ili obrisati SYSTEM tabelspace. Podrazumevano, Oracle baza podataka postavlja sve novo kreirane korisničke tablespacove da budu lokalno upravljani. U bazi podataka sa lokalno upravljanim SYSTEM tabelspace-om, ne možete kreirati tablespacove upravljane rečnikom (koji su zastareli). Međutim, ako izvršimo CREATE DATABASE naredbu ručno i prihvatimo podrazumevane vrednosti, onda je SYSTEM tabelspace upravljan prema rečniku. Možemo migrirati postojeći SYSTEM tabelspace upravljan prema rečniku u format upravljan lokalno.

**SYSAUX tablespace**

SYSAUX tabelspace je pomoćni tabelspace za SYSTEM tabelspace. Pošto je SYSAUX podrazumevani tabelspace za mnoge Oracle baze podataka funkcije i proizvode koji su ranije zahtevali svoje sopstvene tablespace-ove, to smanjuje broj tablespace-ova potrebnih za bazu podataka. Takođe smanjuje opterećenje na SYSTEM tabelspace-u. Prilikom kreiranja ili nadogradnje baze podataka automatski se kreira SYSAUX tabelspace. Tokom normalnog rada baze podataka, nije dozvoljeno da se SYSAUX tabelspace obriše ili preimenuje. Ako SYSAUX tabelspace postane nedostupan, osnovne funkcije baze podataka i dalje će biti operativne. Međutim, funkcionalnosti baze podataka koje koriste SYSAUX tabelspace mogu da ne uspeju ili da rade sa ograničenom sposobnošću.

**Undo tablespace**

Undo tabelspace je lokalno upravljan tabelspace rezervisan za undo podatke koji se upravljaju sistemski. Kao i ostali trajni tablespaces, undo tablespace-ovi sadrže datoteke podataka. Undo blokovi u ovim datotekama su grupisani u ekstenzije.

**Shadow tablespaces**

Shadow tabelspace je bigfile tabelspace namenjen za zaštitu od izgubljenog pisanja u shadow podacima. Zaštita od izgubljenog pisanja u shadow podacima pruža brzo otkrivanje i reagovanje na izgubljeno pisanje. Izgubljeno pisanje podataka bloka se dešava kada I/O podsistem potvrdi završetak pisanja bloka iako pisanje nije izvršeno ili kada prethodna slika bloka prepiše trenutnu sliku. Neotkriveno izgubljeno pisanje može rezultirati oštećenjem podataka jer se netačni podaci mogu koristiti za druge DML transakcije.

Na primer, transakcija može pročitati stare i netačne podatke iz jedne tabele, a zatim ažurirati stotine drugih tabela na osnovu tih podataka. Na taj način, oštećenje podataka može se proširiti kroz celu bazu podataka. Zaštita od izgubljenog pisanja u shadow podacima pruža sledeće koristi:

* Otkriva izgubljeno pisanje pre nego što se koristi za standardne DML, SQL\*Loader konvencionalno učitavanje, direktan put učitavanja i RMAN sigurnosne kopije
* Nije potrebna standby baza podataka, kao što je slučaj kod zaštite od izgubljenog pisanja uvedene u Oracle Database 11g
* Moguće je omogućiti zaštitu od izgubljenog pisanja u shadow podacima za određene tablespacove i datoteke podataka. Nije potrebno pratiti sve podatke
* Moguće je zameniti jedan shadow tabelspace drugim da biste promenili njegovu konfiguraciju ili lokaciju
* Moguće je suspendovati i nastaviti zaštitu od izgubljenog pisanja u shadow podacima za tabelspace ili datoteku podataka
* Moguće je omogućiti ili onemogućiti za celu non-CDB[[4]](#footnote-4) ili PDB [[5]](#footnote-5)sa jednom ALTER DATABASE ... LOST WRITE TRACKING naredbom

Za omogućavanje i onemogućavanje zaštite od izgubljenog pisanja u shadow podacima koristi se ALTER DATABASE naredba. Da bi zaštita od izgubljenog pisanja u shadow podacima štitila određeni tabelspace ili datoteku podataka, moraju se ispuniti sledeći uslovi:

* Moramo omogućiti zaštitu od izgubljenog pisanja u shadow podacima za celu non-CDB ili PDB koristeći ALTER DATABASE ENABLE LOST WRITE PROTECTION naredbu.
* Moramo omogućiti zaštitu od izgubljenog pisanja u shadow podacima za tabelspace ili datoteku podataka koja treba da bude zaštićena koristeći ENABLE LOST WRITE PROTECTION klauzulu. Kada omogućite zaštitu od izgubljenog pisanja u shadow podacima za tabelspace, sve datoteke podataka tog tablespaca su zaštićene, kao i sve datoteke podataka dodate u taj tabelspace.
* Moramo kreirati jedan ili više shadow tablespace-ova koristeći CREATE BIGFILE TABLESPACE naredbu sa LOST WRITE PROTECTION klauzulom.

Oracle Database automatski dodeljuje praćenu datoteku podataka određenom shadow tabelspace-u. Ne možemo odrediti koji shadow tabelspace će se koristiti za određenu datoteku podataka.

Sledeće naredbe data dictionary-ja prate shadow tablespace-ove:

* DBA\_TABLESPACES

Pokazuje koji su tablespace-ovi shadow tablespace-ovi pretragom.

* DBA\_DATA\_FILES.LOST\_WRITE\_PROTECT

Pokazuje da li je zaštita od izgubljenog pisanja omogućena za datoteku podataka.

* USER\_TABLESPACES.LOST\_WRITE\_PROTECT

Pokazuje da li je zaštita od izgubljenog pisanja u shadow podacima uključena za određeni tabelspace. DBA\_DATA\_FILES ne pokazuje da li je zaštita od izgubljenog pisanja uključena za tabelspace: umesto toga, treba pogledati USER\_TABLESPACES.

**Primer 11:** *Konfiguracija zaštite od izgubljenog pisanja*

Ovaj primer omogućava praćenje izgubljenog pisanja u shadow tablespace-ovima za određeni skup tablespaca. U ovom primeru, cilj je da se zaštite tablespace-evi salestbs i hrtbs unutar non-CDB baze podataka. Takođe želimo da zaštitimo datoteku oetbs00.dbf, i samo tu datoteku, unutar tablespaca oetbs. Neophodni koraci:

1. Prijava u bazu podataka kao SYSTEM.
2. Kreiranje jednog shadow tabelspace na sledeći način:

CREATE BIGFILE TABLESPACE shadow\_lwp1

DATAFILE 'shadow\_lwp01\_df' SIZE 10M LOST WRITE PROTECTION;

1. Omogućavanje zaštitu od izgubljenog pisanja za celu bazu podataka na sledeći način:

ALTER DATABASE ENABLE LOST WRITE PROTECTION;

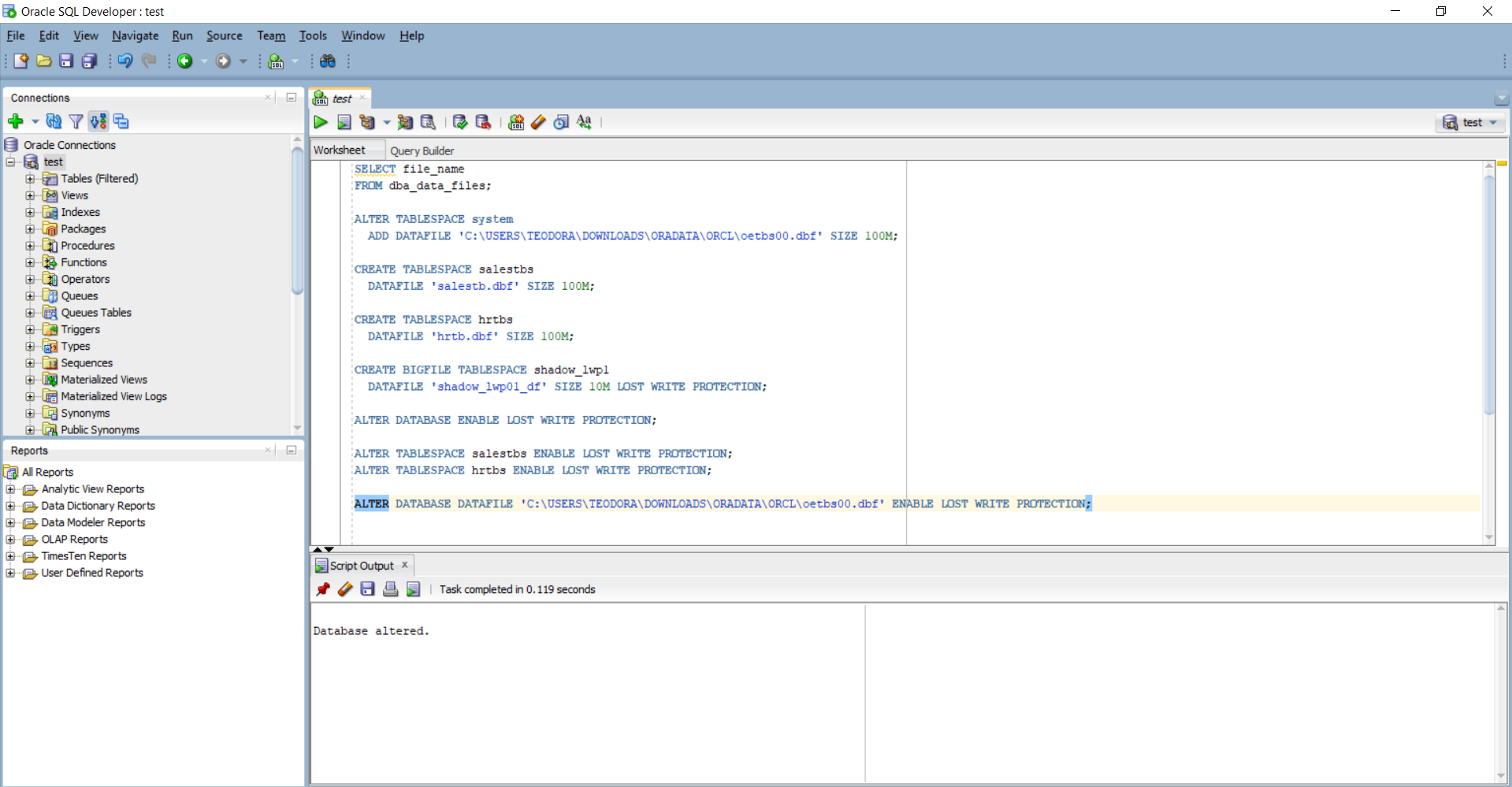
1. Omogućavanje zaštitu od izgubljenog pisanja za tablespace-ove salestbs i hrtbs na sledeći način:

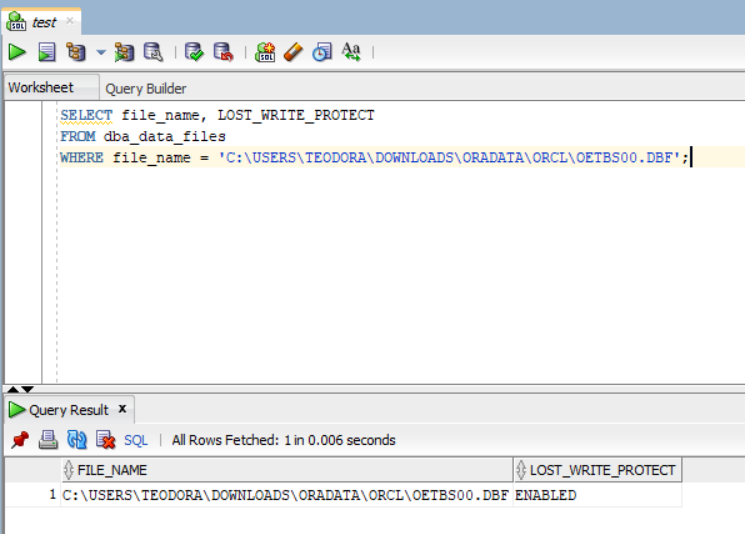
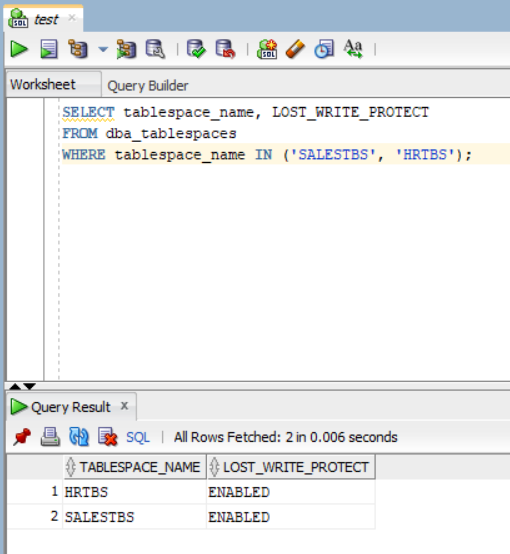
ALTER TABLESPACE salestbs ENABLE LOST WRITE PROTECTION;

ALTER TABLESPACE hrtbs ENABLE LOST WRITE PROTECTION;

1. Omogućavanje zaštitu od izgubljenog pisanja za datoteku podataka oetbs01.dbf na sledeći način:

ALTER DATABASE DATAFILE 'oetbs00.dbf' ENABLE LOST WRITE PROTECTION;



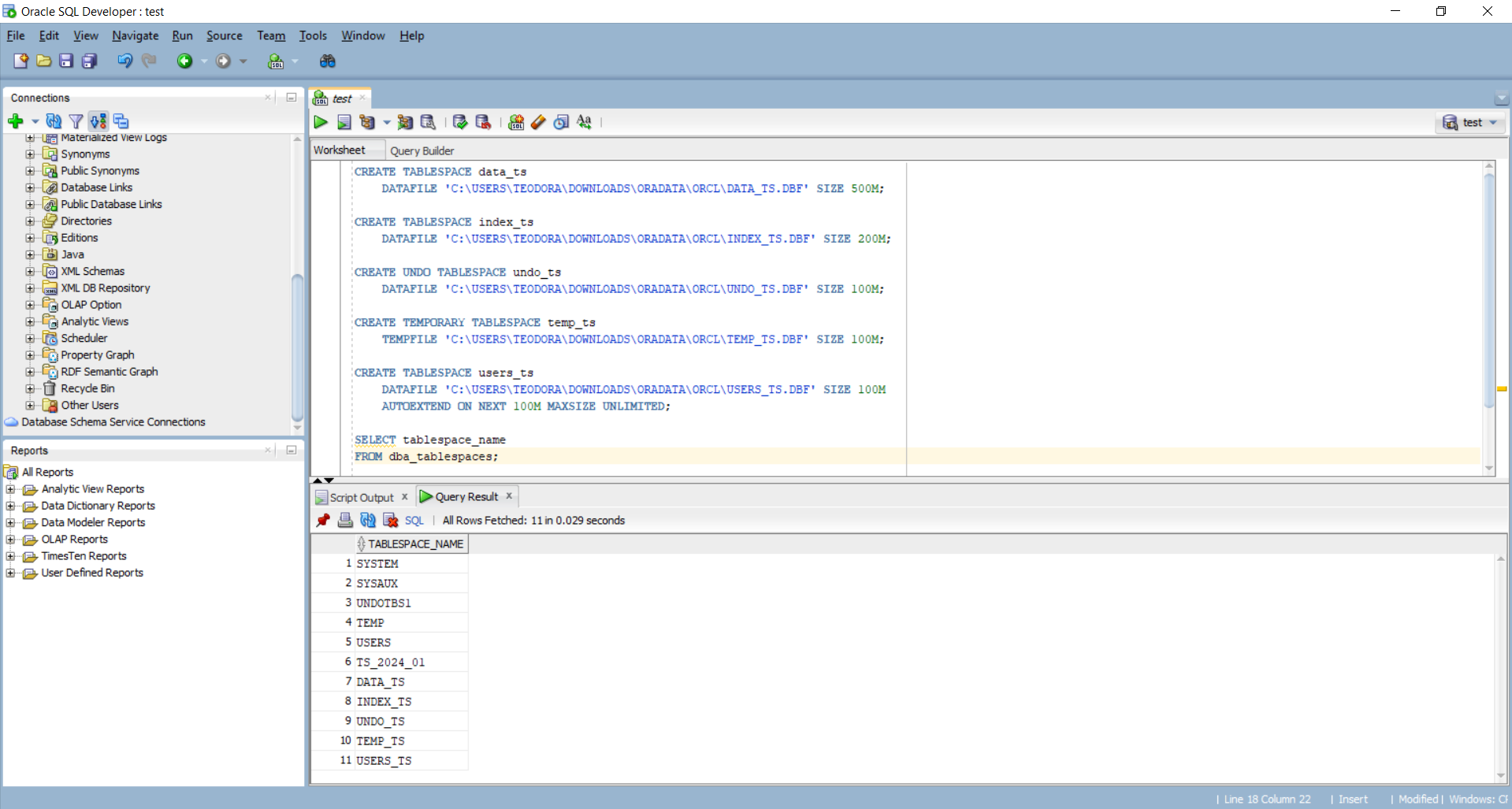
 

**Privremeni tabelspace**

Privremeni tabelspace sadrži privremene podatke koji postoje samo tokom trajanja sesije. U privremenom tabelspace-u ne mogu postojati stalni objekti šeme. Temp datoteka čuva podatke privremenog tabelspace-a. Privremeni tablespaces mogu poboljšati konkurentnost više sortiranja operacija koje ne staju u memoriju. Takođe, ovi tablespace-ovi poboljšavaju efikasnost operacija upravljanja prostorom tokom sortiranja.

**Praktični primeri**

**Primer 12.** Kreiranje tablespace-ova za skladištenje podataka, skladištenje indeksa, upravljanje undo segmenata, privremene podatke i kreiranje tablespace-a sa automatskim proširenjem (sa početnom veličinom od 100MB i automatskim proširivanjem za 100MB kada je potrebno, maksimalna veličina datoteke nije ograničena).



# Zaključak

Ovaj seminarski rad pruža sveobuhvatan pregled interne strukture i organizacije skladištenja podataka u Oracle bazi podataka. Kroz detaljan prikaz fizičkih i logičkih struktura skladištenja, kao i njihovog upravljanja, predstavljeni su ključni elementi koji čine osnovu efikasnog upravljanja podacima u Oracle okruženju. Analiza Oracle ASM-a, datoteka podataka, kontrolnih datoteka i online Redo Log datoteka pruža dublji uvid u načine na koje se podaci fizički organizuju i upravljaju na disku, dok istraživanje logičkih struktura skladištenja, uključujući blokove podataka, ekstenzije, segmente i tablespace-ove, objašnjava kako se podaci logički organizuju unutar baze radi efikasnog upravljanja i pristupa.

Ovaj pregled omogućava bolje razumevanje kako Oracle baza podataka upravlja podacima na nivou skladištenja, što doprinosi optimizaciji performansi i poboljšanju pouzdanosti sistema. Dodatno, pruža osnovu za efikasnije upravljanje prostorom na disku, što je ključno za skalabilnost i optimalno korišćenje resursa. Razumevanje kako se podaci fizički i logički organizuju omogućava administratorima baza podataka da donose odluke o skaliranju, rezervisanju prostora i upravljanju resursima.

Praktični primeri koji su obuhvaćeni u radu ilustruju primenu teorijskih koncepata u stvarnim scenarijima, pružajući konkretne smernice za upravljanje internim strukturama skladištenja podataka u Oracle bazi. Kroz integraciju teorijskih koncepta sa praktičnim primerima, ovaj rad pruža sveobuhvatan pregled pristupa efikasnom upravljanju podacima u Oracle okruženju.

# Literatura

1. Lance Ashdown, Tom Kyte, Joe McCormack; „Oracle Database Architecture“, Oracle

Dostupno: <https://docs.oracle.com/en/database/oracle/oracle-database/18/cncpt/introduction-to-oracle-database.html>

1. Lance Ashdown, Tom Kyte, Joe McCormack; „Physical Storage Structures“, Oracle

Dostupno: <https://docs.oracle.com/en/database/oracle/oracle-database/18/cncpt/physical-storage-structures.html>

1. Lance Ashdown, Tom Kyte, Joe McCormack; „Logical Storage Structures“, Oracle

Dostupno: <https://docs.oracle.com/en/database/oracle/oracle-database/18/cncpt/logical-storage-structures.html>

1. "*Oracle Database 19c Technical Architecture*", Oracle

Dostupno: <https://www.oracle.com/webfolder/technetwork/tutorials/architecture-diagrams/19/pdf/db-19c-architecture.pdf>

1. „Oracle Database Architecture“, OracleTutorial

Dostupno: <https://www.oracletutorial.com/oracle-administration/oracle-database-architecture>

1. Oracle ASM (Automatic Storage Management) je tehnologija za upravljanje podacima u Oracle bazi podataka. [↑](#footnote-ref-1)
2. RMAN (Recovery Manager) je alat za upravljanje sigurnosnim kopijama i obnavljanje podataka u Oracle bazi podataka. [↑](#footnote-ref-2)
3. LOB (Large Object) je tip koji se koristi za čuvanje velikih binarnih ili tekstualnih podataka unutar baza podataka. [↑](#footnote-ref-3)
4. Non-CDB (Non-Container Database) je tradicionalni tip baze podataka u Oracle okruženju, koji se koristio pre pojave Multitenant arhitekture. [↑](#footnote-ref-4)
5. PDB (Pluggable Database) je koncept u Oracle Multitenant arhitekturi koji omogućava kreiranje više izolovanih baza podataka unutar jednog Oracle Container Database (CDB). [↑](#footnote-ref-5)