Lezione 17

Indici e dimensionamento

Ridurre i tempi di ricerca

- In generale le tabelle di un database contengono milioni di righe e vengono lette insieme a molte altre tabelle in istruzioni SQL molto complesse
- Uno dei maggiori problemi dei database è quindi l'ottimizzazione dei tempi di risposta delle ricerche
- Con la definizione degli indici si introducono strutture di dati che velocizzano la ricerca di un valore in una colonna di una tabella

Modalità di ricerca in un insieme

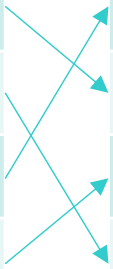
- Se l'insieme non è ordinato l'unica ricerca possibile di un valore è scorrere l'insieme sequenzialmente con complessità temporale n se n è il numero di elementi dell'insieme
- Se l'insieme è ordinato la complessità temporale della ricerca si riduce di molto al valore di $\log_2 n$
- Se la tabella non si presenta ordinata, allora si devono aggiungere ad essa strutture dati ordinate per consentire di velocizzare la ricerca

Gli indici

- Gli indici sono delle strutture dati progettate per individuare velocemente ciò che si cerca, senza dover scorrere tutti i dati uno per uno scorrendo l'intera tabella, una tupla alla volta
- L'idea di base è quella di associare a ogni tupla t_i di una tabella coppie del tipo (k_i,p_i) in cui
  - k_i è il valore di un campo o di un insieme di campi della tupla
  - p_i è la posizione della tupla nella tabella
- Gli indici sono insiemi di coppie (k_i,p_i) tenuti ordinati per i valori di k_i

Esempio

k	p
1001	2
1005	4
1006	1
1020	3



Matricola	Esame	Voto	Data	Docente
1006	Dasi Dati	23	01/11/2020	Rossi
1001	Reti	30	10/03/2019	Bianchi
1020	Informatica	28	08/09/2018	Pica
1005	Reti	18	06/06/2029	Salvo

Indexed Sequential

- Una stessa tabella può avere più indici
- La presenza degli indici comporta
  - maggiore velocità in operazioni di ricerca
  - maggiore lentezza nelle operazioni di inserimento, modifica e cancellazione dovuta alla necessità di tenere ordinati gli indici
    - tanto più lenta quanti più sono gli indici definiti sulla tabella

Quando usare gli indici

- È fondamentale capire quando creare indici
  - se si pensa che in una operazione di inserimento oltre al tempo richiesto per l'inserimento del record vero e proprio, occorre aggiungere il tempo necessario per la gestione di ciascun indice
- In particolare, conviene creare un indice se:
  - la tabella è di grandi dimensioni e la maggior parte delle query recupera spesso meno del 2-4% delle righe;
  - la colonna da indicizzare contiene un elevato numero di valori NULL;
  - la colonna da indicizzare contiene un'ampia varietà di valori;
  - la colonna viene usata spesso in una clausola WHERE o in una condizione di JOIN;
  - la colonna è una FOREIGN KEY e si vuole conservare l'integrità dei dati mediante dei vincoli;
- Può invece non convenire creare un indice se:
  - la colonna non viene usata spesso come condizione di ricerca di una query;
  - la tabella viene aggiornata molto di frequente;
  - la colonna indicizzata viene usata come parte di un'espressione;
  - la tabella è di piccole dimensioni o la maggior parte delle query recupera spesso più del 2-4% delle righe.

Creazione degli indici

- In modo automatico se nella definizione di una tabella sono definiti i vincoli
  - PRIMARY KEY
  - UNIQUE
  - sono creati automaticamente indici per gli attributi con il vincolo
- In modo esplicito (o manuale) con un comando CREATE

```
CREATE [UNIQUE] [BITMAP] INDEX nome_indice
ON nome_tabella (nome_colonna [, nome_colonna] …);
```

- Per eliminare un indice creato in modo esplicito

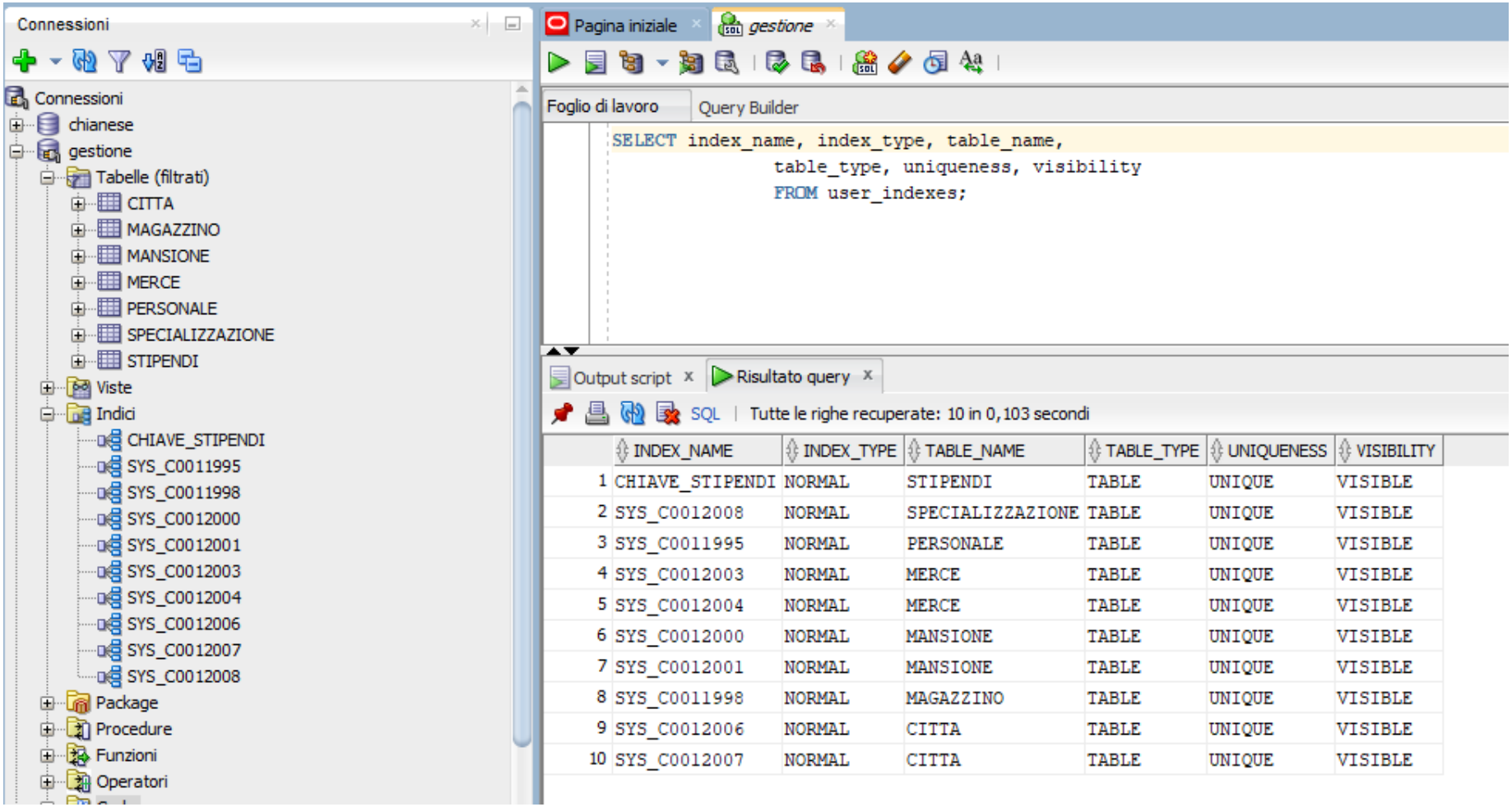
```
DROP INDEX nome_indice;
```

Si noti che il DROP di una tabella elimina anche gli indici ad essa associati

Quando creare gli indici

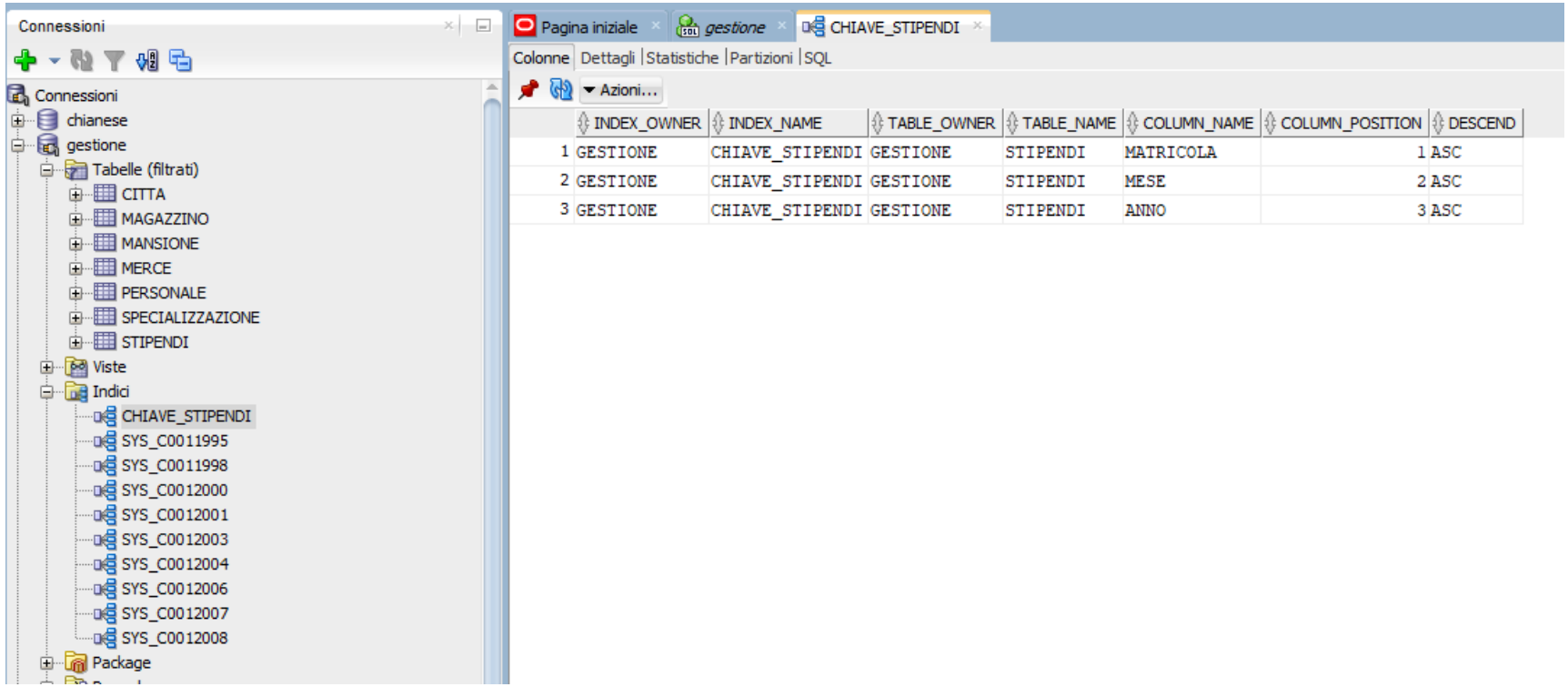
- Gli indici impliciti si attivano insieme alla creazione della tabella
- Gli indici espliciti possono essere creati in un qualsiasi momento della vita della base dati
  - se vengono creati quando il data base è già popolato per migliorare le prestazioni
    - ha un tempo di costruzione che è tanto più lungo quanto più è popolata la tabella a cui è associato

L’elenco degli indici

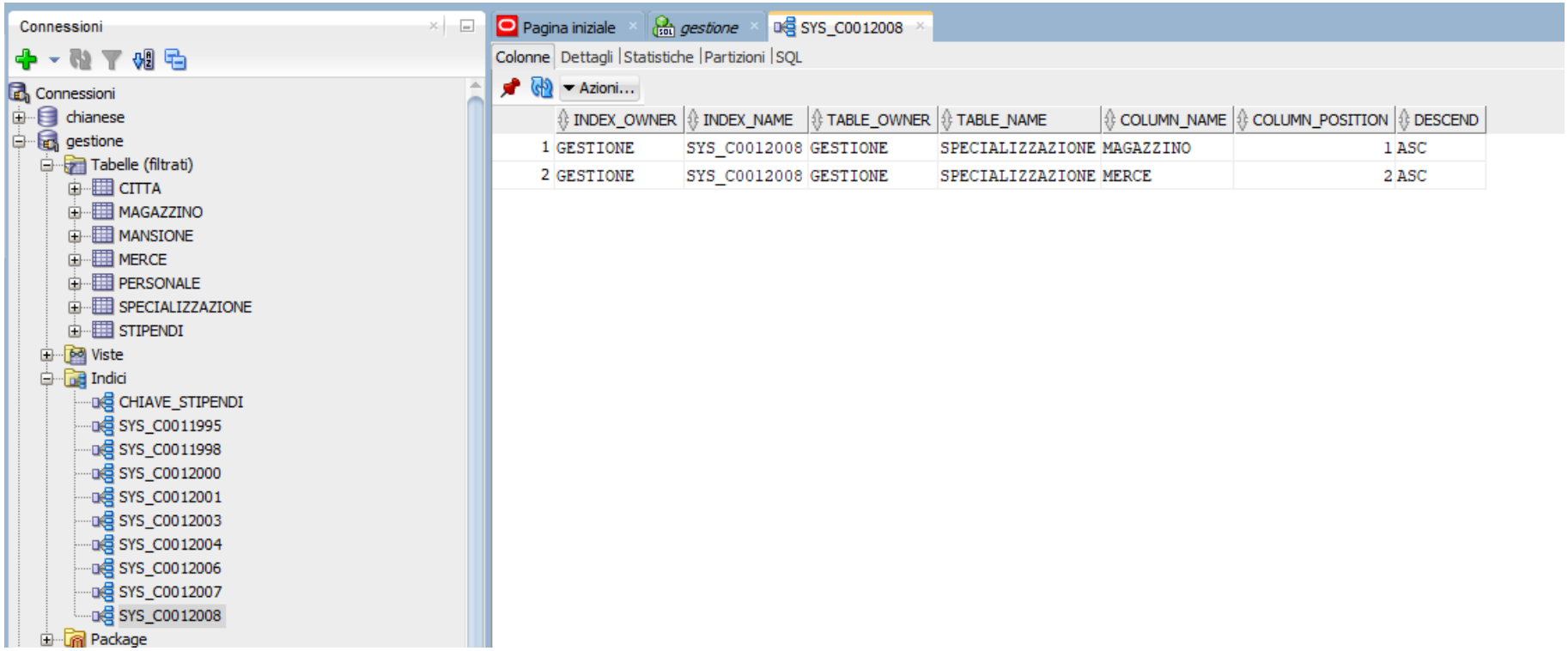


	INDEX_NAME	INDEX_TYPE	TABLE_NAME	TABLE_TYPE	UNIQUENESS	VISIBILITY
1	CHIAVE_STIPENDI	NORMAL	STIPENDI	TABLE	UNIQUE	VISIBLE
2	SYS_C0012008	NORMAL	SPECIALIZZAZIONE	TABLE	UNIQUE	VISIBLE
3	SYS_C0011995	NORMAL	PERSONALE	TABLE	UNIQUE	VISIBLE
4	SYS_C0012003	NORMAL	MERCE	TABLE	UNIQUE	VISIBLE
5	SYS_C0012004	NORMAL	MERCE	TABLE	UNIQUE	VISIBLE
6	SYS_C0012000	NORMAL	MANSIONE	TABLE	UNIQUE	VISIBLE
7	SYS_C0012001	NORMAL	MANSIONE	TABLE	UNIQUE	VISIBLE
8	SYS_C0011998	NORMAL	MAGAZZINO	TABLE	UNIQUE	VISIBLE
9	SYS_C0012006	NORMAL	CITTA	TABLE	UNIQUE	VISIBLE
10	SYS_C0012007	NORMAL	CITTA	TABLE	UNIQUE	VISIBLE

Gli indici associati a una tabella



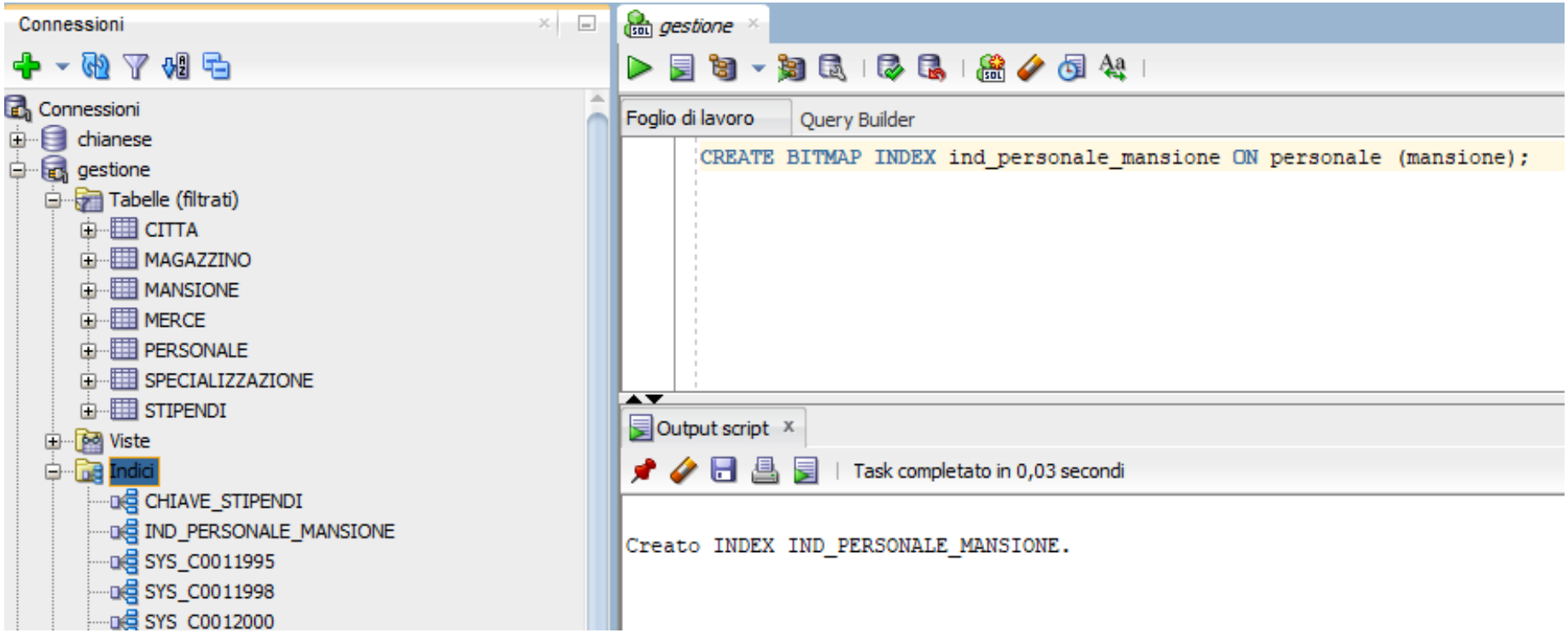
Altro esempio di chiave composta



- Per vedere quali e quanti indici sono presenti in una base dati

```
SELECT index_name, index_type, table_name,
       table_type, uniqueness, visibility
       FROM user_indexes;
```

Indice bitmap



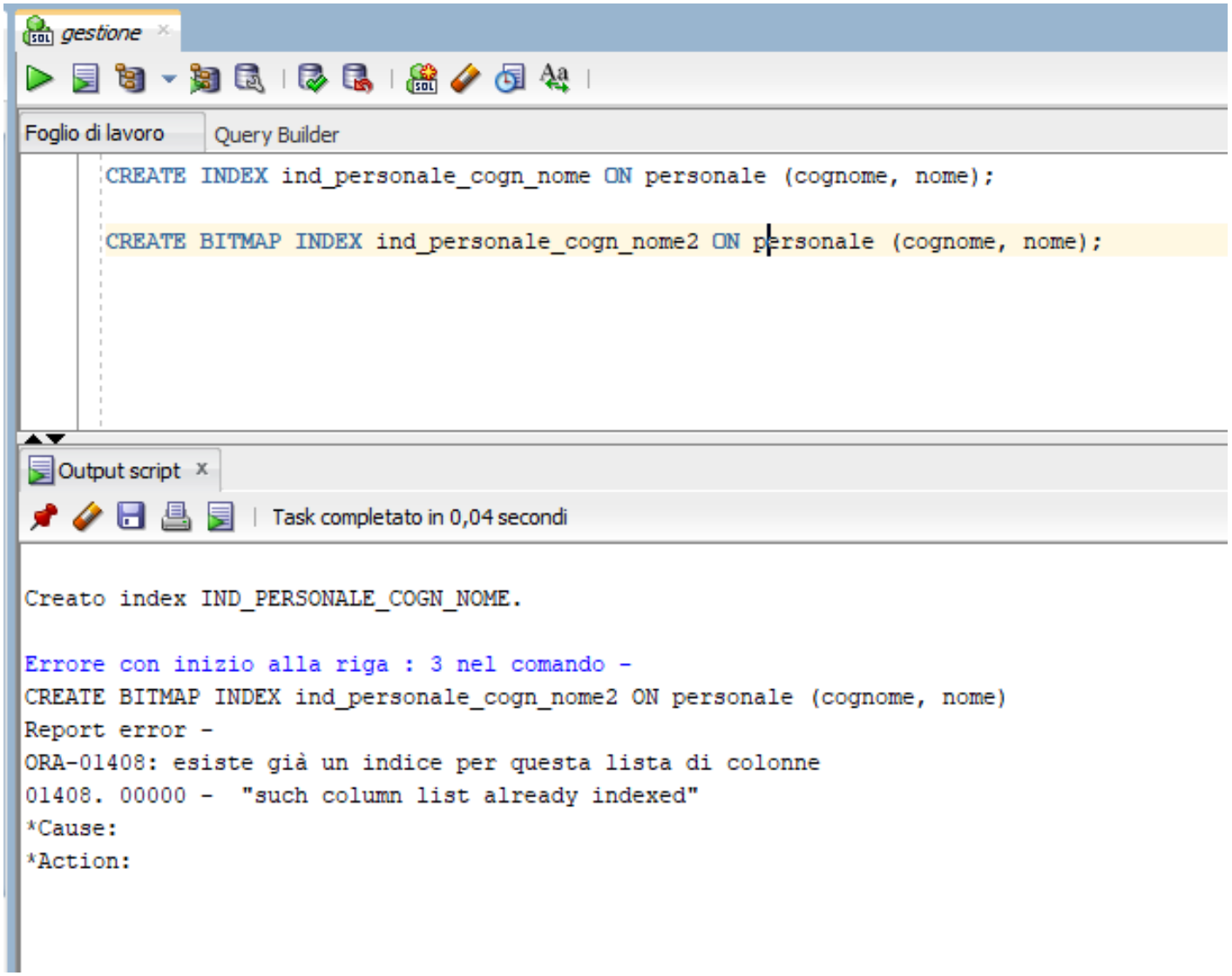
Indici invisibili

- Oracle ha introdotto la funzionalità *indici invisibili* utile in diverse situazioni
  - Un indice invisibile è un indice gestito dal database ma ignorato dall'ottimizzatore
  - L'indice invisibile è un'alternativa alla cancellazione di un indice
- Un uso dell'indice invisibile è verificare le prestazioni senza di esso prima di cancellarlo per evitare di doverlo ricostruire di nuovo

Gli indici di oracle

- Oracle dispone di due tipologie di indici ognuna adatta a un particolare tipo di dato o a un particolare tipo di accesso
  - Indici B+Tree
  - Indici Bitmap

Gestione indici



Gli indici invisibili

gestione | IND_PERSONALE_COGN_NOME

Foglio di lavoro | Query Builder

```
ALTER INDEX ind_personale_cogn_nome invisible;

SELECT index_name, index_type, table_name,
       table_type, uniqueness, visibility
       FROM user_indexes;
```

Output script | Risultato query

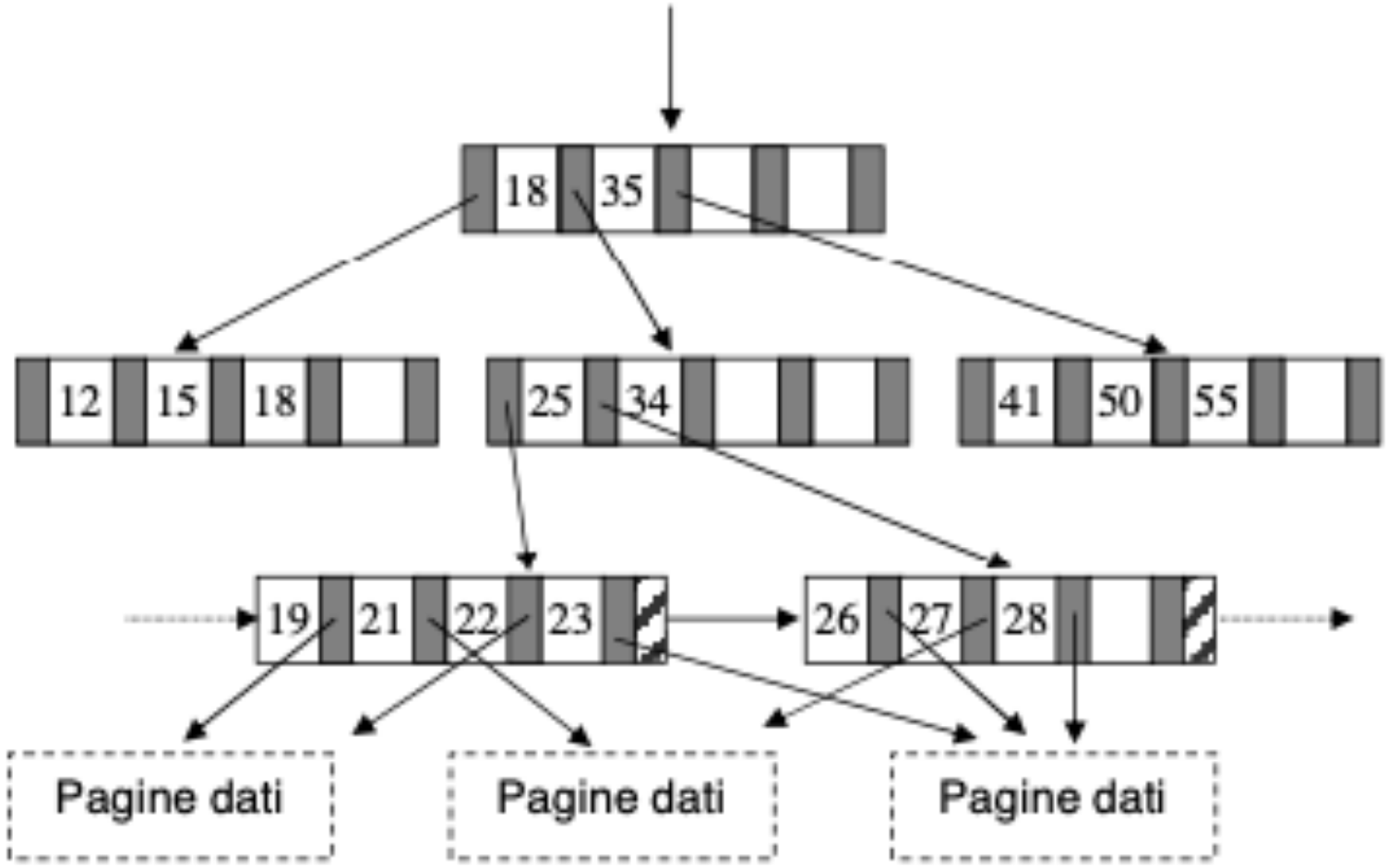
SQL | Tutte le righe recuperate: 12 in 0,065 secondi

	INDEX_NAME	INDEX_TYPE	TABLE_NAME	TABLE_TYPE	UNIQUENESS	VISIBILITY
1	CHIAVE_STIPENDI	NORMAL	STIPENDI	TABLE	UNIQUE	VISIBLE
2	SYS_C0012008	NORMAL	SPECIALIZZAZIONE	TABLE	UNIQUE	VISIBLE
3	IND_PERSONALE_COGN_NOME	NORMAL	PERSONALE	TABLE	NONUNIQUE	INVISIBLE
4	SYS_C0011995	NORMAL	PERSONALE	TABLE	UNIQUE	VISIBLE
5	IND_PERSONALE_MANSIONE	BITMAP	PERSONALE	TABLE	NONUNIQUE	VISIBLE
6	SYS_C0012003	NORMAL	MERCE	TABLE	UNIQUE	VISIBLE
7	SYS_C0012004	NORMAL	MERCE	TABLE	UNIQUE	VISIBLE
8	SYS_C0012000	NORMAL	MANSIONE	TABLE	UNIQUE	VISIBLE
9	SYS_C0012001	NORMAL	MANSIONE	TABLE	UNIQUE	VISIBLE
10	SYS_C0011998	NORMAL	MAGAZZINO	TABLE	UNIQUE	VISIBLE
11	SYS_C0012006	NORMAL	CITTA	TABLE	UNIQUE	VISIBLE
12	SYS_C0012007	NORMAL	CITTA	TABLE	UNIQUE	VISIBLE

I B+-Tree

- Un indice **B+-Tree** per un attributo *c* di una relazione *R*
  - è un albero bilanciato che consente accessi associativi alle tuple di *R* in base ai valori della chiave *c*
  - Le foglie dell'albero, collegate tra loro in sequenza, contengono i puntatori ai record (*RID - Row Identifier*)
  - mentre i nodi interni costituiscono una mappa per consentire una rapida localizzazione delle chiavi.

Un albero ordinato per chiavi numeriche



- Il B+-Tree è una strutture bilanciata, ossia garantisce che l'altezza *h* dell'albero sia sempre costante per tutti i percorsi dalla radice alle foglie
  - nell'esempio h è uguale a tre
- Per ricercare un particolare valore *v*
  - si discende l'albero in base alla mappa dei puntatori
  - tramite la foglia individuata si accede al blocco dati in cui sono memorizzati i record che presentano il valore di chiave *v*
- Per eseguire una ricerca su un intervallo di valori [*a*, *b*] è sufficiente scendere nell'albero utilizzando *a* come valore di chiave, quindi seguire la sequenza di nodi foglia fino a che non si incontra un valore superiore a *b*

Gli indici bitmap

- Un indice bitmap su un attributo è composto da una matrice di bit
  - con tante righe quante sono le tuple della relazione
  - con tante colonne quanti sono i valori distinti dell'attributo
- Il bitmap (*i*,*j*) è posto a TRUE se nella tupla *i*-esima è presente il valore *j*-esimo

Esempio di indice BITMAP

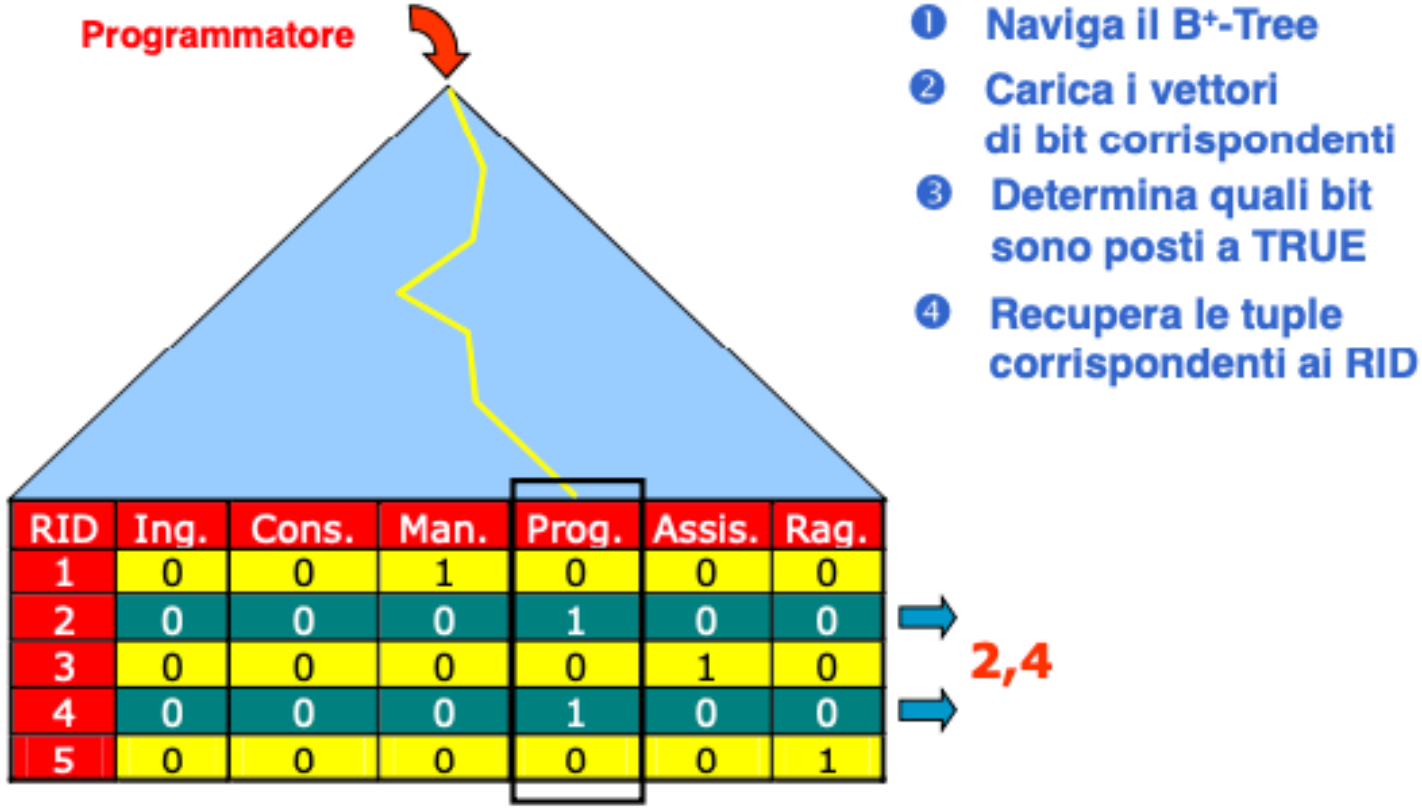
- Esempio di indice sul campo posizione della tabella impiegati
  - che può assumere i seguenti valori: Ingegnere, Consulente, Manager, Programmatore, Assistente, Ragioniere

RID	Ing.	Cons.	Man.	Prog.	Assis.	Rag.
1	0	0	1	0	0	0
2	0	0	0	1	0	0
3	0	0	0	0	1	0
4	0	0	0	1	0	0
5	0	0	0	0	0	1

L'impiegato corrispondente al RID 1 è un Manager

Implementazione dei bitmap

- Normalmente i bitmap sono realizzati con B+-Tree le cui foglie contengono vettori di bit invece di RID



Caratteristiche degli indici BITMAP

- Lo spazio richiesto su disco è ridotto
- L'I/O è molto basso poichè sono letti solo i vettori di bit necessari
- Sono ottimi per interrogazioni che non richiedono l'accesso ai dati
- Permettono l'utilizzo di operatori binari per l'elaborazione dei predicati
- Gli indici bitmap sono adatti ad attributi con una bassa cardinalità poichè ogni nuovo valore distinto di chiave richiede un altro vettore di bit

Gestione degli spazi

L’architettura del DBMS ORACLE

- L’architettura del DBMS ORACLE è di tipo client-server
  - in cui il DBMS è un processo *server* che eroga servizi
- L’architettura del server ORACLE si compone di tre componenti fondamentali:
  - una o più *istanze*
  - uno o più *processi listener*
  - uno o più *oracle database*, ognuno legato ad una data istanza

L’istanza

- Una *istanza* è un insieme di processi e di strutture dati allocate in memoria, che creano il meccanismo di accesso ai dati e ne permettono la gestione
- L’istanza deve essere attiva e risiedere, insieme alle proprie strutture dati, in una specifica area di memoria denominata *SGA* (*System Global Area*)
- Nella SGA si trovano le strutture dati necessarie al funzionamento del DBMS

Alcune strutture dati presenti nella SGA

- La Shared pool è un'area usata per memorizzare gli statement SQL utilizzati recentemente per reperire informazioni in maniera più veloce
- Il Database buffer è un'area usata per memorizzare e gestire i blocchi di dati oggetto delle elaborazioni.
  - La dimensione del buffer ha il maggiore impatto sulle prestazioni globali del sistema di base di dati
- Il Redo Log buffer è un'area che contiene le informazioni relative ai blocchi dati modificati in seguito ad operazioni sulle base di dati. Essa è fondamentale per la ricostruzione della base di dati a valle di rollback o malfunzionamenti
- Le Data and Library caches sono aree di cache della shared pool utilizzate per contenere i dati in transito da e verso la base di dati (data) e le istruzioni SQL e PL/SQL in esecuzione (library)

Alcuni processi di un'istanza ORACLE

- Dispatcher process (D000), Query processor (QP) e Server process
  - Implementano parte delle funzionalità di un Gestore delle Query e di un Gestore degli Accessi
  - Il primo è il processo responsabile dello scheduling delle richieste utente.
  - Il secondo è il processore delle query
  - Il terzo è un processo dedicato che nasce per ogni utente od applicazione che si connette al DBMS. Ogni Server process utilizza un'area di memoria dedicata detta Program Global Area (PGA) per servire le richieste utente contenente le informazioni sulla sessione utente corrente
- Database writer (DBW0), Process monitor (PMON) e User process (UP)
  - Implementano le funzionalità tipiche del Gestore della Memoria e del Gestore dei File
  - Il primo è responsabile della scrittura dei blocchi di dati dal database buffer nei datafile
  - Il secondo è responsabile del rilascio delle risorse allocate da un eventuale processo utente fallito
  - Il terzo è il processo che in corrispondenza delle richieste utente preleva blocchi dati richiesti dai data file e li porta nel buffer qualora non siano già presenti nella SGA
- Log writer (LGWR), Checkpoint process (CKPT), Recoverer process (REC0) e Archiver process (ARC0)
  - Implementano le funzionalità tipiche di un Gestore dell'Affidabilità
  - Il primo è responsabile della scrittura, degli eventi che si susseguono in una sessione di lavoro, dal Redo log buffer (memoria) nel log file (disco)
  - Il secondo è responsabile dell'aggiornamento dello stato del database, per quanto concerne i datafile, ogni volta che viene memorizzato permanentemente un record nel database (garantisce la sincronizzazione dei dati)
  - Il terzo è responsabile della risoluzione delle transazioni fallite in ambiente distribuito
  - Il quarto è responsabile del salvataggio automatico delle copie dei redo log in un'area di memoria stabile (Archived Log Files) specificata dall'amministratore del database.
- System monitor (SMON)
  - Implementa le funzionalità del Gestore dell'Integrità
  - Il processo è responsabile del controllo della consistenza e dell'integrità del database, se necessario inizia un recovery del database, quando esso è attivo
- Lock process (LMS)
  - Implementa le funzionalità tipiche del Gestore della Concorrenza
  - È responsabile della gestione dei lock sulle transazioni in ambiente distribuito

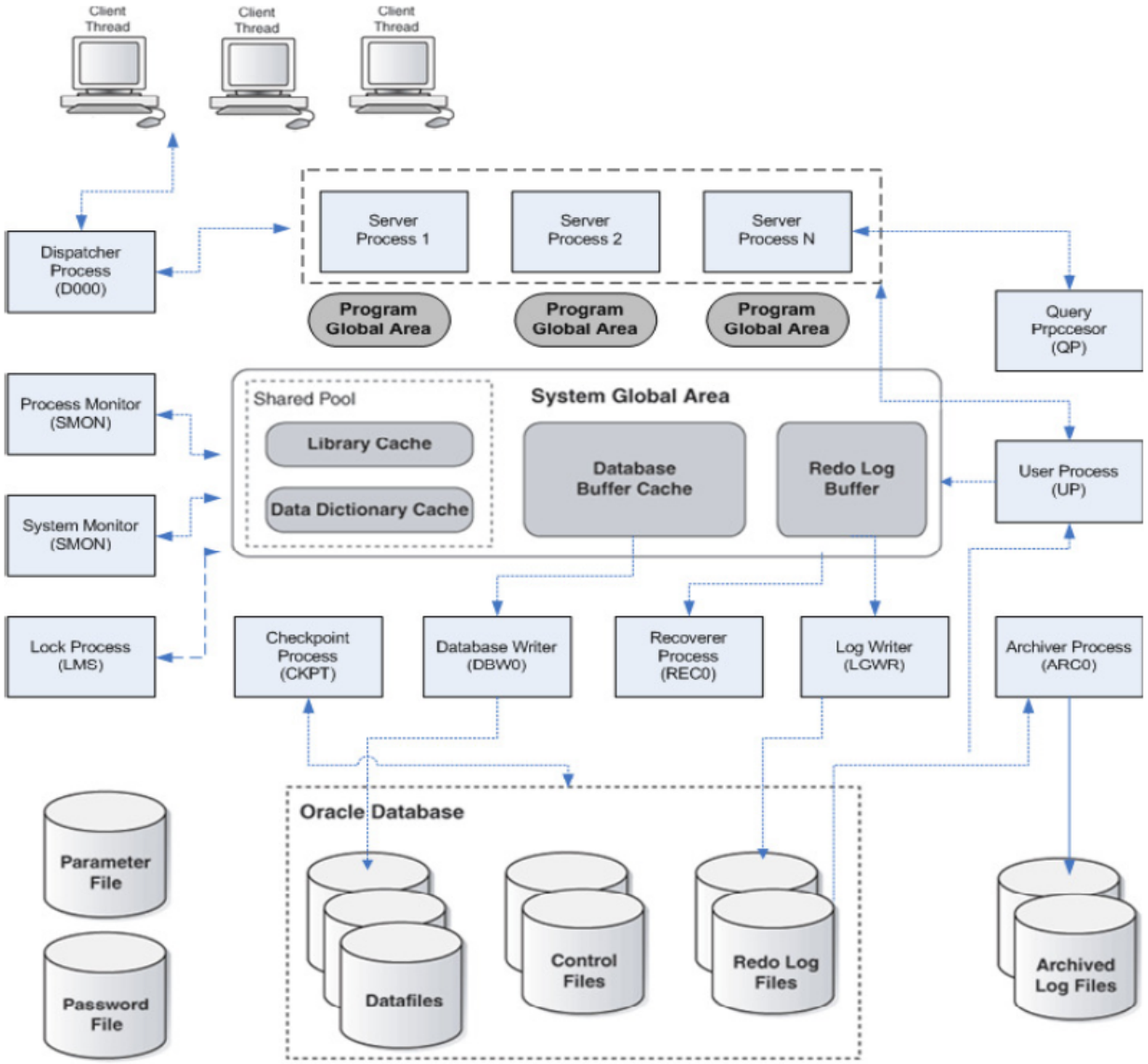
Il listener

- Il listener è il processo asincrono che si occupa di gestire la comunicazione fra i processi client e le corrispondenti istanze ORACLE
- Un processo listener è caratterizzato da un *nome*, dal *database oracle* che serve e da una *porta di ascolto*
- La configurazione dei *listner* attivi su una base di dati ORACLE e` presente in un particolare file chiamato *LISTNER.ORA*

Un database Oracle

- È una collezione di dati trattata come una singola unità che ha una sua struttura logica e fisica
- Un oracle database è in composto da tre tipi di file
  - I Data file che contengono i dati attuali della base di dati, le informazioni sugli indici associati alle tabelle ed il catalogo
    - un datafile è associato a un solo database
    - i datafile hanno spazi di memoria fissati automaticamente
    - uno o più datafile formano una unità logica chiamata tablespace
  - Il Redo Log file che contiene informazioni, scritte in un record, per l'eventuale recovery della base di dati in caso di rollback.
  - Il Control file che contiene le indicazioni per l'uso della base di dati e informazioni sull'istanza (nome database, percorso file di log, numero sequenziale corrente del redo log, informazioni su checkpoint, ecc.)
- In ambito distribuito il database viene univocamente identificato
  - da un nome detto SID (System Identificator)
  - dall'indirizzo IP o nome a dominio dell'host che lo ospita
  - dalla porta di accesso su cui è in ascolto il processo listener

Uno sguardo all'architettura



L’esecuzione di un comando SQL

- Il comando passa dal dispatcher al server process dedicato che controlla se l’istruzione è già contenuta nella library cache
  - per il riuso dell’albero di decodifica e del piano di esecuzione
  - altrimenti è decodificata per far generare dal processore delle query , se corretta, un piano di esecuzione che con l’albero di decodifica viene memorizzato nella library cache
- Per gli oggetti interessati dall’istruzione viene controllato se i corrispondenti blocchi di dati si trovano nel buffer del database
  - In caso negativo il processo user legge i blocchi di dati dai files fisici al buffer del database
  - per recuperare spazio i blocchi di altri oggetti utilizzati meno recentemente vengono portati su disco dal processo DBWR.
- Le modifiche delle tuple interessate dal comando avvengono nel buffer del database.
  - Prima però una loro immagine viene scritta sui *segmenti di rollback* dal processo DBWR
- Il buffer redo-log viene riempito a causa delle modifiche ai blocchi e il processo LGWR scrive sui file le voci in esso presenti
- Se viene lanciato il commando di *rollback* i blocchi modificati vengono sovrascritti dai blocchi originali conservati nei segmenti di rollback.
- Se l’utente invia un comando di commit
  - lo spazio occupato dai blocchi nel segmento di rollback viene liberato e reso quindi disponibile per altre tran- sazioni
  - i blocchi modificati nel buffer del database vengono sbloccati in modo che gli altri utenti possano accedervi
- La fine della transazione viene registrato nei file redo-log
  - I blocchi modificati vengono scritti su disco solo se si rende necessario spazio per altre transazioni

L’organizzazione della memoria

- In un qualsiasi architettura DBMS si è soliti distinguere la struttura logica da quella fisica
- A livello più basso o fisico si trovano i file gestiti dal sistema operativo ospite
- Le aree logiche di memorizzazione (dette spazi) gestiscono gli oggetti secondo la loro strutturazione logica

Le aree logiche di Oracle

- Oracle è organizzato nelle seguenti aree logiche:
  - database
  - tablespace
  - segmenti
  - extent
  - block

- Un database è formato da diversi tablespace
  - un tablespace da diversi segmenti
    - un segmento da diversi extent
      - un extent da diversi block

Le aree

- Tablespace
  - Tutti gli oggetti del database vengono logicamente memorizzati in uno o più tablespace
- Segmenti
  - quando un oggetto (per esempio una tabella) viene creato gli viene assegnato un segmento
- Extent
  - la più piccola unità logica di memorizzazione che può essere allocata per un oggetto di database
  - consiste in una sequenza contigua di blocchi (*block*) di dati
  - se la dimensione di un oggetto cresce (per esempio, per l'inserimento di tuple nelle tabelle) gli viene allocato un extent aggiuntivo
- Block
  - corrisponde ad un numero di byte (multiplo del buffer usato dal sistema operativo) scelto dal DBA in fase di creazione della base di dati

Il processo di creazione di un database

1. Dimensionamento fisico della base di dati
   - sia globale che a livello dei singoli oggetti
   - calcolo dello spazio di memorizzazione (*storage*) richiesto su disco
2. Creazione del database
3. Definizione delle politiche di sicurezza
4. Creazione degli utenti/ruoli
5. Creazione degli oggetti (tabelle, indici) e definizione dei vincoli
6. Creazione di un'istanza (popolamento della base di dati)

Per vedere i tablespace

```
SELECT tablespace_name, file_name,
       bytes / 1024/ 1024 MB
FROM dba_data_files;
```

Creazione di un nuovo schema

- Definizione dello spazio complessivo

  CREATE TABLESPACE

- Creazione utente

  CREATE USER

Dimensionamento globale

CREATE TABLESPACE Nome Tablespace
DATAFILE Nome file **SIZE** Dimensione
{, **DATAFILE** Nome file **SIZE** Dimensione};

CREATE TABLESPACE spazio_per_magazzino
DATAFILE ‘mag1.dbf’ **SIZE** 500k,
DATAFILE ‘mag2.dbf’ **SIZE** 500k;

- Il primo della dimensione richiesta dalla base di dati
- Il secondo della stessa dimensione utilizzabile nel caso si saturi il primo

Associazione utente amministratore

CREATE USER username
IDENTIFIED BY password
[**DEFAULT TABLESPACE** tablespace]
[**QUOTA** {**size** | **UNLIMITED**} **ON** tablespace]
[PROFILE profile] [**PASSWORD EXPIRE**]
[**ACCOUNT** {**LOCK** | **UNLOCK**}];

CREATE USER mag_dba
IDENTIFIED BY quellachevuoi
DEFAULT TABLESPACE spazio_per_magazzino
QUOTA UNLIMITED ON spazio_per_magazzino;

GRANT ALL PRIVILEGES TO mag_dba;

I parametri

- DEFAULT TABLESPACE
  - Identifica la memoria dove saranno allocate tabelle e viste
- QUOTA
  - Specifica lo spazio che l'utente può usare
- PASSWORD EXPIRE
  - Se si vuole forzare il cambio password al primo accesso
- ACCOUNT {LOCK | UNLOCK}
  - Per blocccare o sbloccare l'accesso da parte dell'utente

La lista degli utenti

```
SELECT username, default_tablespace, profile,
authentication_type
FROM dba_users
WHERE account_status = 'OPEN';
```

La lista dei privilegi

- **SELECT** * **FROM** session_privs **ORDER BY** privilege;

Per allocare una tabella

CREATE TABLE table_name(

...

) **TABLESPACE** tablespace_name;

Dimensionamento di una tabella

```
CREATE TABLE nome_tab (
                    …
) STORAGE (INITIAL 100K NEXT 50K
                MINEXTENTS 1 MAXEXTENTS 50
                PCTINCREASE 5);
```

Parametri

- INITIAL e NEXT
  - specificano la dimensione del primo extent e di quelli successivi
- MINEXTENTS
  - specifica il numero totale di extents allocati quando il segmento viene creato
  - permette di allocare una grande quantità di spazio quando un oggetto viene creato, anche se lo spazio disponibile non e` contiguo
- MAXEXTENTS
  - specifica il numero massimo ammissibile di extents
- PCTINCREASE
  - specifica la percentuale con la quale ogni extent dopo il secondo cresce rispetto all'extent precedente
  - il valore di default è 50 che indica che l'extent successivo al secondo cresce solo del 50%