CSCI317 Database Performance Tuning

Persistent Storage Structures

Dr Janusz R. Getta

School of Computing and Information Technology -
University of Wollongong

Persistent Storage Structures

Outline

Storage pyramid

Action	Time in nanoseconds (1 ns = 10-9 second)
L1 cache reference	0.5
Branch mispredict	5
L2 cache reference	7
Main memory reference	10
Mutex lock/unlock	25
Transmission of 2Kb over 1 Gb/s net	20,000 20 microsec
SSD random read	150,000 150 microsec
HDD random read	10,000,000 10 millisec
Send a package over wide area net	150,000,000 150 millisec

Storage pyramid

CPU registers

Cache memory

Main memory

Solid State Drive (SSD)

Hard Disk Drive (HDD)

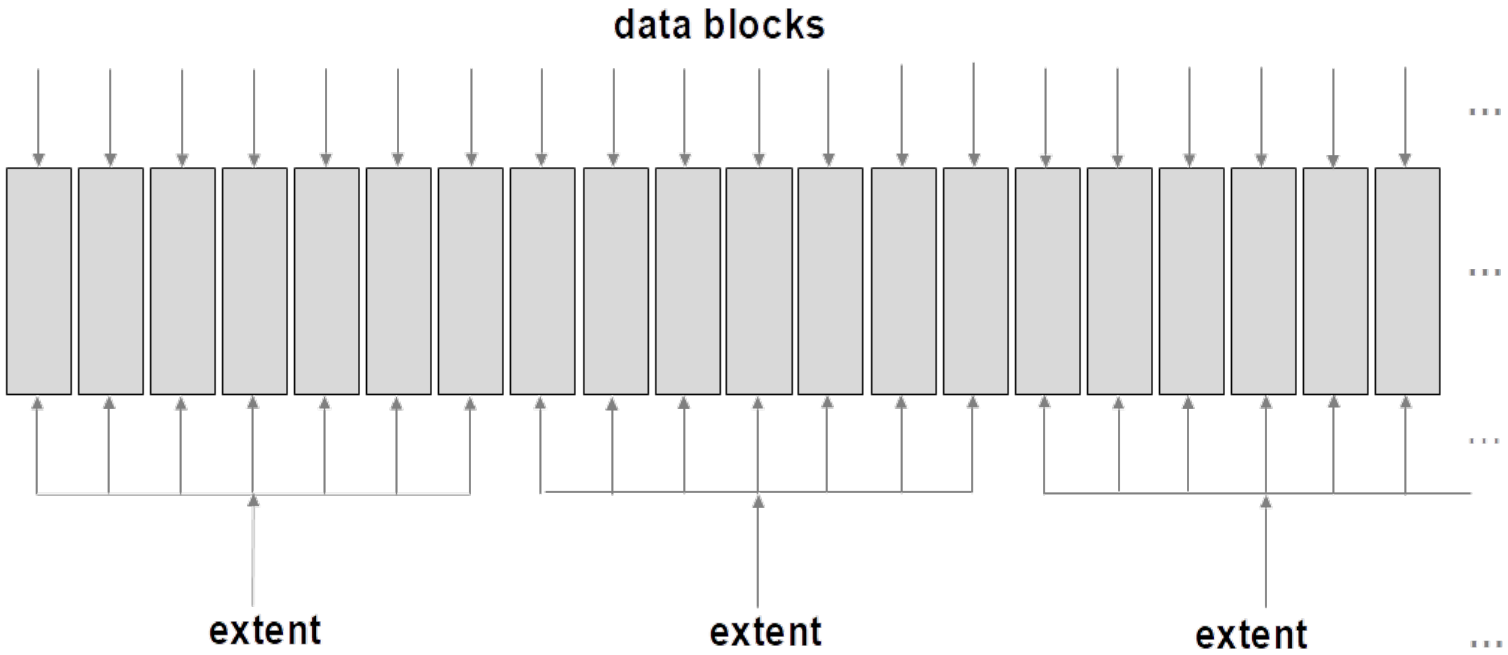
Optical disk/Magnetic tape

Shelved optical disk/shelved magnetic tape

Persistent Storage Structures

Outline

Logical model of persistent storage



A data block is a contiguous sequence of `2 Kbytes`, or `4 Kbytes`, or `8 Kbytes`, or `16 Kbytes`, or `32 Kbytes`

A data block is identified by block address

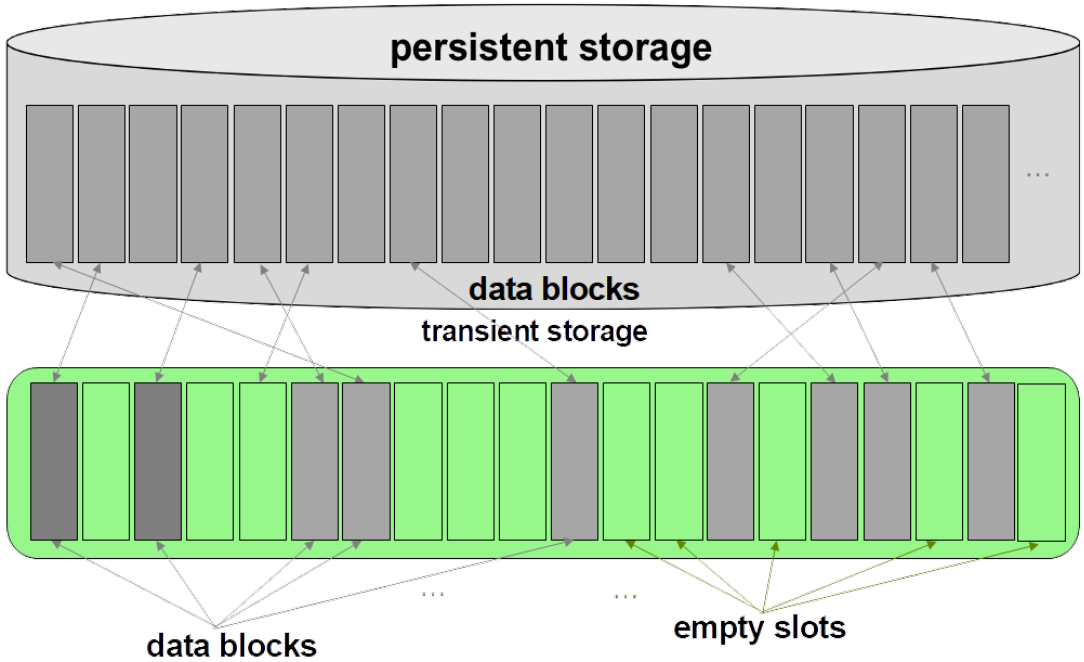
An extent is a contiguous sequence of data blocks

Persistent Storage Structures

Outline

Persistent storage versus transient storage

A data buffer pool (data cache) is an area of transient storage that contains the blocks transmitted from persistent storage



A data buffer pool is used to reduce a number of transmissions from persistent storage to transient storage and reverse

Persistent storage versus transient storage

Whenever a data block is read by a database application and such block is not in a buffer pool then it is transmitted from persistent storage

Whenever a data block is written by a database application then write operation is performed on block in transient storage and such block is not immediately written to persistent storage

Multiple reads and writes on the same data block are performed on the block located in transient storage

Hit ratio = (number of read/write operations on data buffer pool)/ (total number of read/write operations)

In a well tuned system hit ratio ≥ 0.9

It means that out of 100 read/write operations at least 90 must be performed on data buffer pool

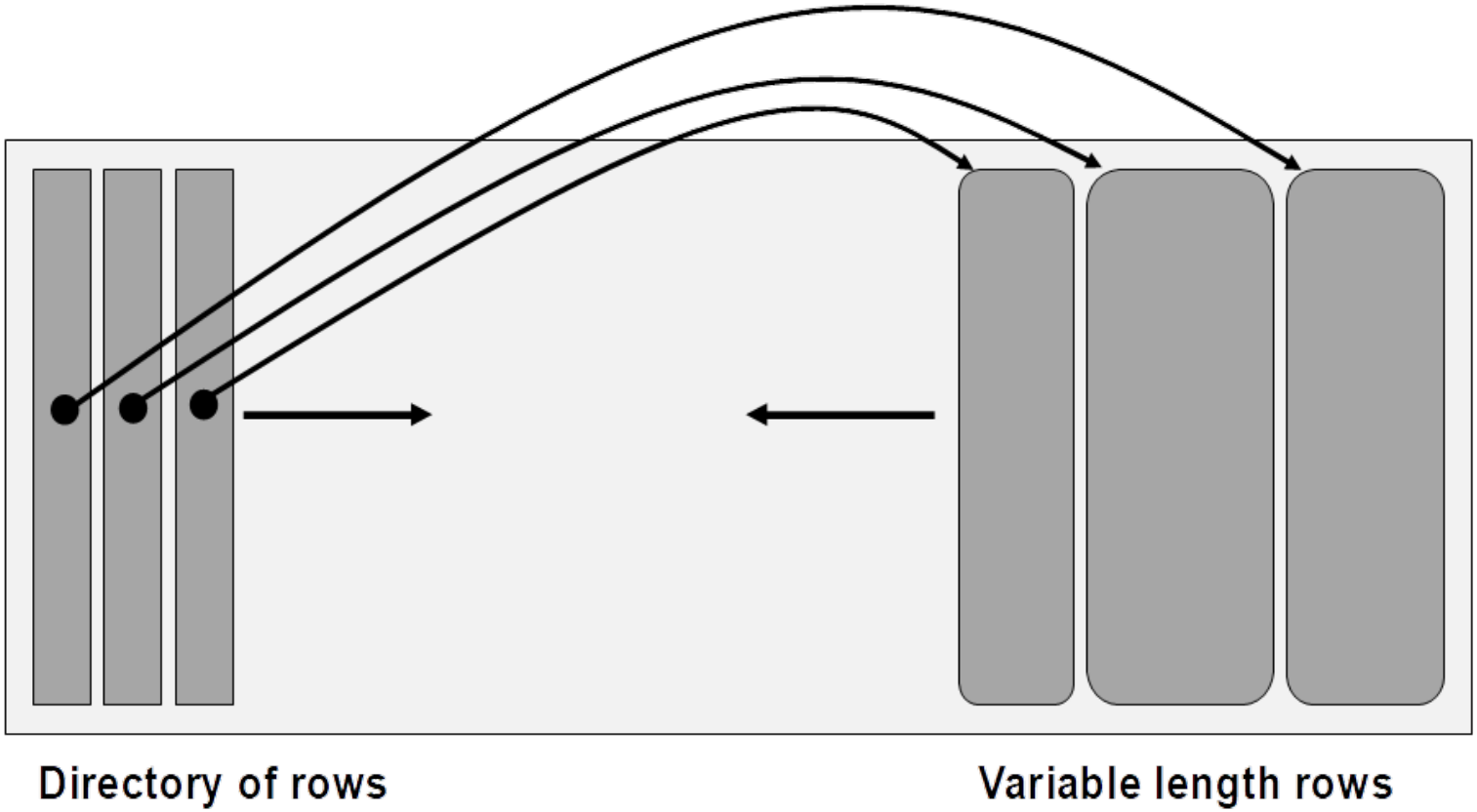
Persistent Storage Structures

Outline

- Storage pyramid
- Logical model of persistent storage
- Persistent storage versus transient storage
- Internal data block format
- Storage structures
- Segments
- Tablespaces
- Distribution of data objects
- Manipulation on tablespaces

Internal data block format

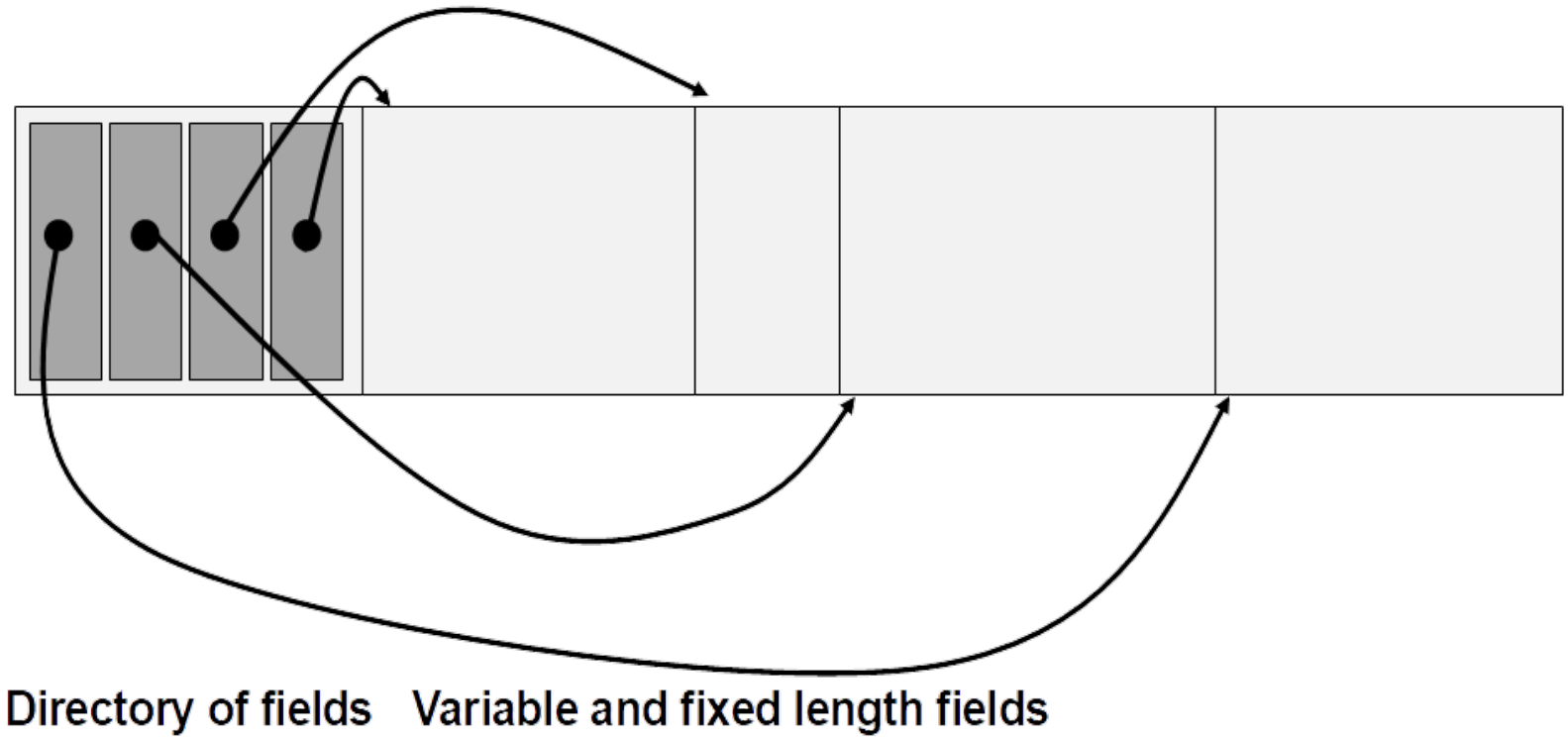
Row-oriented implementation or relational tables



Column-oriented implementation of relational tables (discussed later)

Internal data block format

Directory-oriented implementation of rows in a relational table



Persistent Storage Structures

Outline

Storage structures

Data block

- A smallest unit of disk space allocation

Data block size

- Multiplicity of operating system block size, e.g. 2K, 4K, 8K,16K, 32K

Extent

- An extent consists of specific number of contiguous data blocks (default 5 blocks, minimum 2 blocks).

Segment

- A segment consists extents allocated for a specific type of data structure and stored in the same tablespace
- Each relational table is stored in its own data segment, each index is stored in its own index segment, etc

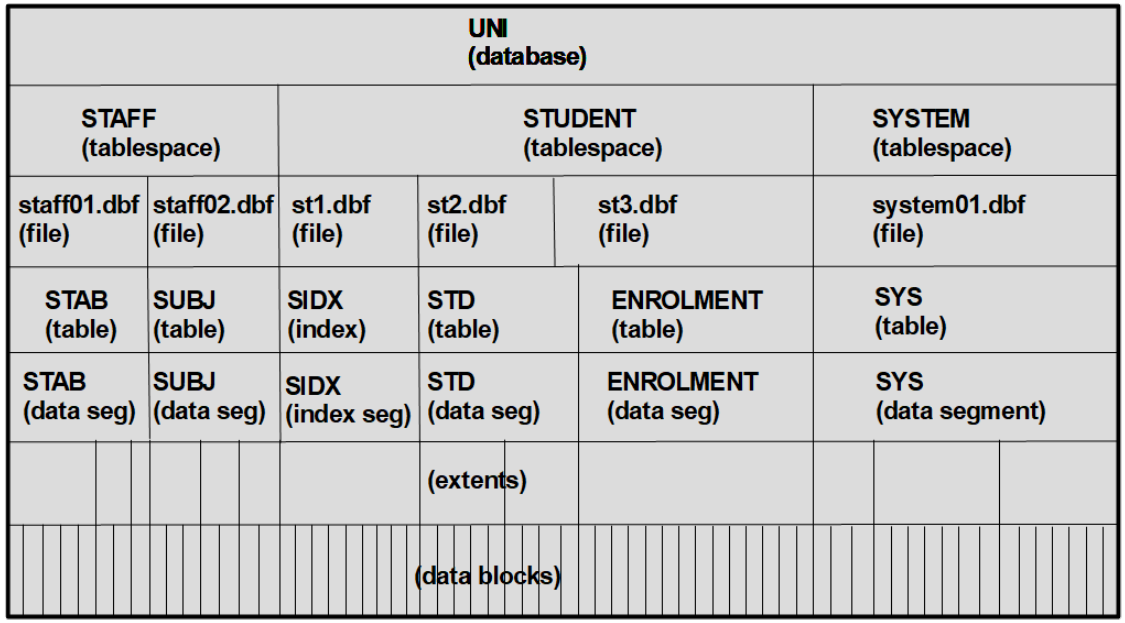
Storage structures

Tablespace

- A logical storage component of a database. At a physical level a tablespace consists of files. At a logical level a tablespace consist of segments (relational tables, indexes, materialized views, etc)

Data file

- A physical storage component of a database



Persistent Storage Structures

Outline

Segments

Data segments

- Tables, materialized views, clusters

Index segments

- Indexes

Rollback/UNDO segments

- Segments that contain data blocks updated by database transactions

Temporary segments

- Segments required for intermediate stages of SQL processing

Persistent Storage Structures

Outline

Tablespaces

`SYSTEM` tablespace

- `SYSTEM` tablespace contains all data dictionary tables
- `SYSTEM` tablespace contains stored procedures and triggers
- `SYSTEM` tablespace is automatically created when a database is created

Performance related observations:

`SYSTEM` tablespace must be located on the fastest persistent storage device available

`SYSTEM` tablespace must must not share its persistent storage device with other tablespaces

`SYSTEM` tablespace must not be used as a user tablespace

Tablespaces

`UNDO` tablespace

- `UNDO` tablespace contains information about old contents of data blocks
- `UNDO` tablespace is needed to perform rollback operations

Performance related observations:

`UNDO` tablespace should be large enough to allow for rollback of the transactions that modify a lot of data

A size of `UNDO` tablespace determines the depth of transaction stack, i.e. a limit in processing of "old" transactions

Tablespaces

`TEMP` tablespace

- `TEMP` tablespace contains temporary segments
- `TEMP` tablespace is used for sorting, hashing, processing of SQL statements
- `TEMP` tablespace is automatically created when a database is created

Performance related observations:

`TEMP` tablespace should be large anough to allow for fast processing of sorting and hashing

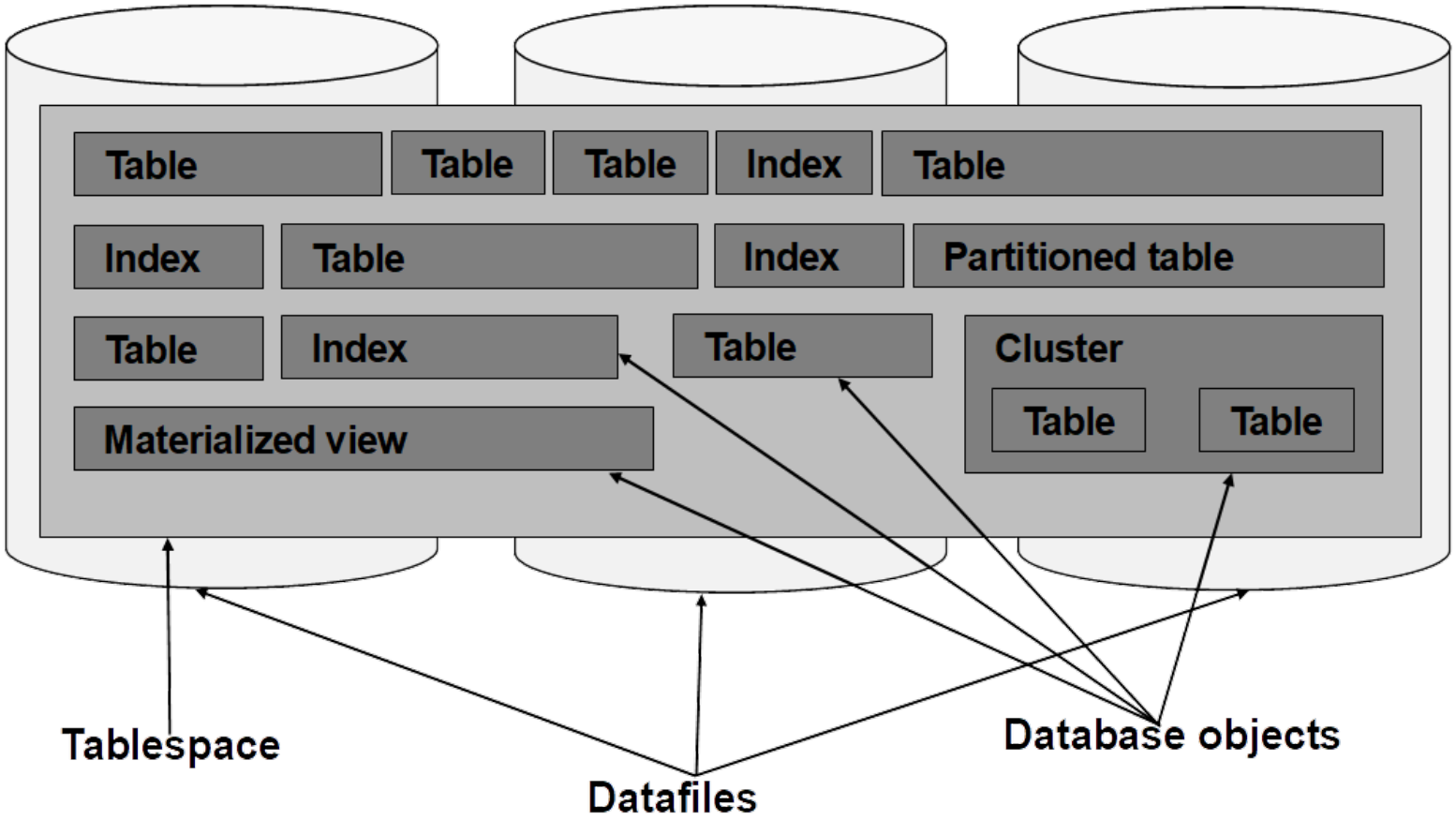
`TEMP` tablespace cannot not be used as user tablespace

Persistent Storage Structures

Outline

- Storage pyramid
- Logical model of persistent storage
- Persistent storage versus transient storage
- Internal data block format
- Storage structures
- Segments
- Tablespaces
- Distribution of data objects
- Manipulation on tablespaces

Distribution of database objects



Distribution of database objects

Performance related observations:

Distribution of data objects (relational tables, indexes, ..., etc) has a very important impact on performance

Data objects that are used together should be located on different devices to allow for simultaneous transmission of data from persistent storage to main memory

For example, two relational tables that are frequently joined should be located on the different persistent storage devices

Data objects that are the most frequently used should be located on the fastest persistent storage devices

It is important to consider a balance between a size and frequency of use, for example a small and frequently used relational table should be located on a device that is faster than very large and very rarely used relational table

Persistent Storage Structures

Outline

- Storage pyramid
- Logical model of persistent storage
- Persistent storage versus transient storage
- Internal data block format
- Storage structures
- Segments
- Tablespaces
- Distribution of data objects
- Manipulation on tablespaces

Manipulations on tablespaces

Creating tablespaces

Creating a new tablespace

```
CREATE TABLESPACE EXAMPLE
  DATAFILE 'C:\ORACLE\ORA90\DB\EXMP01.DBF'
  SIZE 100M;
```

Altering tablespaces

Altering an existing tablespace

```
ALTER TABLESPACE EXAMPLE
  RENAME TO NEW_EXAMPLE;
```

Enlarging tablespaces (1)

Enlarging a tablespace

```
ALTER TABLESPACE EXAMPLE
  ADD DATAFILE 'E:\ORACLE\ORADATA\TBS_03.DBF'
  SIZE 3M;
```

Enlarging tablespaces (2)

Enlarging a tablespace

```
ALTER DATABASE
  DATAFILE 'E:\ORACLE\ORADATA\TBS_03.DBF'
  RESIZE 200M;
```

Manipulations on tablespaces

Shrinking tablespaces (1)

Shrinking a tablespace

```
ALTER DATABASE
  DATAFILE 'E:\ORACLE\ORADATA\TBS_03.DBF'
  RESIZE 20M;
```

Shrinking tablespaces (2)

Shrinking a tablespace

```
ALTER TABLESPACE EXAMPLE
  DROP DATAFILE 'E:\ORACLE\ORADATA\TBS_03.DBF'
```

Dynamically resizing tablespaces

Creating a new autoextensible tablespace

```
CREATE TABLESPACE EXAMPLE
  DATAFILE 'C:\ORACLE\ORA90\DB\EXMP01.DBF'
  AUTOEXTEND ON
  SIZE 100M;
```

Making a database file extensible

```
ALTER DATABASE
  DATAFILE 'E:\ORACLE\ORADATA\TBS_03.DBF'
  AUTOEXTEND ON;
```

Manipulations on tablespaces

Dropping empty tablespaces

Dropping an empty tablespace

```
DROP TABLESPACE EXAMPLE;
```

Dropping nonempty tablespaces

Dropping a nonempty tablespace

```
DROP TABLESPACE EXAMPLE INCLUDING CONTENTS;
```

Dropping nonempty tablespaces with referential integrity constraints

Dropping a nonempty tablespace with referential integrity constraints

```
DROP TABLESPACE EXAMPLE
  INCLUDING CONTENTS
  CASCADE CONSTRAINTS
```

Dropping nonempty tablespaces with referential integrity constraints and files

Dropping a nonempty tablespace with referential integrity constraints and data files

```
DROP TABLESPACE EXAMPLE
  INCLUDING CONTENTS
  CASCADE CONSTRAINTS
  AND DATAFILES;
```

References

Cookbook, How to create, to manage, and to analyze tablespaces, how to analyze allocation of extents in tablespaces ?

Cookbook, How to defragment persistent storage at tablespace level, at segment level, and at extent level ?

Lightstone, S., Teorey T., Nadeau T., Physical Database Design, The Database Professional's Guide to Exploiting Indexes, Views, Storage, and More, Morgan Kaufmann Publishers, 2007, chapter 1

Kyte, T., Expert Oracle Database Architecture, 9i and 10g Programming Techniques and Solutions, APress, 2005, chapter 10 (Available from "Other Resources" section on Moodle)