

networking - PDF Understanding Summary



LectureStorage.pdf
PDF

read and understand this pdf carefully. just understand and remember it, dont say anything

Understood. I have carefully read and understood the entire PDF document titled "**Space Management Tablespace Demonstration of Implementation**" by Professor Abu Raihan Mostofa Kamal. I will retain the concepts and technical details internally for further use. Let me know when you're ready to proceed or if you have any questions based on it.

explain page 3 and page 5 in details and clear explanation

Here's a detailed and clear explanation of **Page 3** and **Page 5** from the PDF titled "*Space Management Tablespace Demonstration of Implementation*":

● Page 3: Storage: Tables and Objects in General

◆ What it says:

So far, users of the database system don't really know **where** exactly data (like tables or other database objects) are physically stored or how that storage is maintained. Now the focus shifts to showing **how a database designer can control** this storage **more precisely**.

◆ Explanation in Simple Terms:

When you're using a database and you create a table or insert some data, you typically don't think about **where** that data is saved on the system—whether it's in one file, spread across multiple disks, or how the system manages it behind the scenes.

But **as a database administrator (DBA) or designer**, you **need control** over storage:

- You may want to know **which files** are storing what data.
- You may want to improve **performance, backup, or space usage**.
- You might want to assign **specific objects to specific storage locations**.

So, this chapter begins to teach how you can **manually manage and assign storage** for database objects using **tablespaces**.

● Page 5: Tablespace, Datafiles, and Objects (Cont.)

This page explains the relationships and differences between four key concepts:

- **Databases**
- **Tablespaces**
- **Datafiles**
- **Segments**

◆ Concept Breakdown:

1. Databases and Tablespaces:

- An **Oracle database** is made up of **one or more tablespaces**.
- Each **tablespace** is a **logical storage unit**—it organizes how data is stored.
- Besides storage, the database also includes **background processes** (like log writing, process management, etc.).
- So, the actual **data** of the database is stored **inside tablespaces**.

2. Tablespaces and Datafiles:

- A **tablespace** consists of **one or more datafiles**.
- A **datafile** is a file on the disk that **physically stores the data** (e.g., a `.dbf` file in Oracle).
- Therefore, tablespaces are **logical containers**, and datafiles are **physical files**.

3. Databases and Datafiles:

- All of a database's data is **ultimately stored in datafiles**.
- Example:
 - A simple database → 1 tablespace → 1 datafile.
 - A complex database → multiple tablespaces → each having multiple datafiles.
 - If 3 tablespaces have 2 datafiles each, that's **6 datafiles total**.

4. What is a Tablespace's Role?

- It **bridges logical and physical storage**:
 - Logical: How we organize tables, indexes, rollback segments.
 - Physical: Actual files on disk.
- Tablespaces store database **objects** like:
 - Tables
 - Indexes
 - Rollback segments (used to undo changes or recover data after failure).

♦ **Important Concept: Rollback Segment**

- A rollback segment stores **before-images** of data:
 - Used to **undo** transactions if needed.
 - Helps in **recovery** after a system crash.
 - Essential for maintaining **data consistency**.

✅ **Summary in One Line:**

- **Page 3** introduces the idea that users don't know how storage works, but designers can control it using **tablespaces**.
- **Page 5** breaks down how Oracle databases store data logically (tablespaces) and physically (datafiles), and how these interact with database objects like tables and rollback segments.

Let me know if you want a diagram to visualize these relationships!

explain tablespace in understandable manner

Sure! Here's a simple and clear explanation of **tablespace**:

● **What is a Tablespace?**

A **tablespace** is a **logical storage unit** in a database. Think of it like a **folder** in your computer where files are stored, but in a **database system**, it stores **tables, indexes, and other data objects**.

Why Tablespace is Needed:

- To **organize data** better.
 - To **control where data is stored** (e.g., on which drive or disk).
 - To manage **space usage** and **performance**.
-

How it Works (Simple Analogy):

Imagine a **bookshelf**:

Concept	Analogy	Real Meaning
Database	Library	Whole database system
Tablespace	Bookshelf section	A logical area to group related data
Datafile	Book	Actual file that stores the data
Table	Page inside the book	The actual table/data saved in a datafile

So, a **tablespace** is like a **section of the bookshelf** where you decide to store a specific category of books (e.g., math books). Similarly, you store specific **tables or indexes** in a tablespace.

Tablespace is Connected To:

1. **Datafiles** (physical files on disk)
2. **Database Objects** (like tables, indexes, rollback segments)

When you create a table, you can specify **which tablespace** it should be stored in. That way, you control **where it goes**.

✓ Key Points:

- A **tablespace** groups data logically.
 - It is made up of **datafiles** (which store actual data).
 - You can assign users or tables to specific tablespaces.
 - Helps manage **storage, security, and performance** better.
-

Let me know if you'd like a diagram or a live SQL example too!

now explain datafiles and objects

Certainly! Here's a simple and clear explanation of **Datafiles** and **Objects** in the context of Oracle databases:

🔵 What are Datafiles?

🧱 Definition:

A **datafile** is a **physical file** on the disk that stores the **actual data** of a database.

When you create a **tablespace**, you have to assign one or more **datafiles** to it. These datafiles are where Oracle actually **writes and reads the data**.

📁 Example:

- You create a tablespace called `student_data` .

- You assign it a datafile like:

```
/u01/oracle/data/student_data01.dbf
```

👉 That `.dbf` file is the **datafile** where Oracle will **physically store** the data from the `student_data` tablespace.

📌 Key Features of Datafiles:

- Each tablespace must have at least one datafile.
 - They are created and managed by the DBA.
 - Can be **extended** in size if more space is needed.
 - Datafiles can be stored on **different disks** for performance or backup planning.
-

🔵 What are Database Objects?

🧩 Definition:

Database **objects** are the **structures you create inside a database** to store or organize data.

💡 Common Examples:

Object Type	What it does
Table	Stores rows and columns of data
Index	Speeds up search operations
View	Virtual table based on a query
Sequence	Generates auto-increment numbers
Synonym	Alias for other objects

Object Type	What it does
Rollback Segment	Stores old data for undo/recovery

These **objects** are stored inside a **tablespace**, which is connected to datafiles. So ultimately, objects are saved **physically inside datafiles**, but **organized logically using tablespaces**.

Relationship Summary:

pgsql

Database

- └── **Tablespace** (Logical)
 - └── Datafile (Physical)
 - └── Stores **Database** Objects (**Tables**, Indexes, etc.)

Quick Recap:

- **Datafiles** = Physical files that hold actual data on disk.
- **Objects** = Tables, indexes, views, etc., which store and manage data inside the database.
- Objects are **stored in tablespaces**, and tablespaces are made up of **datafiles**.

Let me know if you'd like this in diagram form!

[explain page 8](#)

Explanation of Page 8: Blocks, Extents, and Segments (Oracle Storage Management)

This page presents a visual concept (Figure 2) and explains **how Oracle manages storage internally**. Let's break it down with clear definitions and examples:

Oracle Storage Levels (Bottom to Top)

Oracle organizes its storage **hierarchically** — from small to big:

1. Data Block

- **Smallest unit** of storage in Oracle.
- Also called: *Oracle block*, *logical block*, or *page*.
- Each data block maps to a **specific number of bytes** on the disk (e.g., 2KB, 4KB, 8KB).
- A **table's rows** are stored in these blocks.

Example:

If you insert data into a table, it is first stored in a data block. Think of it like a "box" that holds a few rows of a table.

2. Extent

- A **group of contiguous data blocks**.
- Oracle allocates storage to database objects (like tables or indexes) **in extents**.
- When a table needs more space, Oracle adds a new extent to it.

Example:

If each data block is 8KB and an extent has 10 blocks, then the extent size is 80KB.

3. Segment

- A **set of extents** allocated to a single database object.
- One table = One segment.
- When a table grows, Oracle adds more **extents** to its **segment**.

Example:

If you create a table `employees`, Oracle creates a **segment** for it. As you insert more rows, Oracle adds **extents** (each containing data blocks) to the segment.

Overall Storage Structure:

SCSS

Segment

└── Extent 1

| └── Data Block 1, Data Block 2, ...

└── Extent 2

 └── Data Block 3, Data Block 4, ...

Summary in Simple Words:

Term	Meaning	Example
Block	Smallest unit of storage	Like a single page in a book
Extent	Group of blocks	A chapter of pages
Segment	Whole storage for a table or object	The entire book for one table

- Oracle **allocates one extent at a time** when creating objects.
- When space runs out in that extent, **another extent is added** automatically.

Let me know if you want me to draw this in a diagram or write practice questions!

explain rest of the pages

Here's a clear and detailed explanation of the **remaining pages (9 to 14)** from the PDF **"Space Management Tablespace Demonstration of Implementation"**:

Page 9: Mapping Between User, Object, and Tablespace (Step 1)

Step 1: Create a Tablespace

sql

```
CREATE TABLESPACE mytspace  
DATAFILE '/u02/oracle/data/lmtbsb01.dbf' SIZE 50M  
EXTENT MANAGEMENT LOCAL AUTOALLOCATE;
```

Explanation:

- `CREATE TABLESPACE mytspace` : Creates a new logical storage unit named `mytspace` .
- `DATAFILE ... SIZE 50M` : Assigns a physical file of 50 MB to the tablespace.
- `EXTENT MANAGEMENT LOCAL` : Oracle internally manages extents for the tablespace.
- `AUTOALLOCATE` : Oracle automatically decides how much space to allocate for each extent (starting from 64KB).

Key Concept:

You're telling Oracle: "Create a new storage space called `mytspace` , and physically store it in this file with smart space allocation."

Page 10: Step 2(a) – Create User and Assign Tablespace

sql

```
CREATE USER iutlearner  
IDENTIFIED BY test123  
DEFAULT TABLESPACE mytspace;
```

Explanation:

- `CREATE USER iutlearner` : A new database user named `iutlearner` is created.
- `IDENTIFIED BY test123` : The user's password is set to `test123` .

- `DEFAULT TABLESPACE myt space` : Any object this user creates (like tables) will be stored in `myt space` unless specified otherwise.

Page 11: Step 2(b) – Create Table and Assign Tablespace

sql

```
CREATE TABLE students (  
  ID NUMBER PRIMARY KEY,  
  Name VARCHAR2(50),  
  DOB DATE,  
  Program VARCHAR2(30)  
) TABLESPACE NEW_TBSPACE;
```

Explanation:

- This creates a table `students` with 4 columns.
- The `TABLESPACE NEW_TBSPACE` clause **overrides the user's default** and stores this table in a different tablespace (`NEW_TBSPACE`).

Extra Notes:

- You can use the `DBA_FREE_SPACE` view to check available space in tablespaces.
- Use `ALL_TABLES` to view all tables and their assigned tablespaces.

Page 12: Step 3 – Check Free Space in Tablespaces

sql

```
SELECT TABLESPACE_NAME,  
       SUM(BYTES)/1024/1024/1024 "FREE SPACE(GB)"  
FROM   DBA_FREE_SPACE  
GROUP BY TABLESPACE_NAME;
```

Explanation:

- This query tells you **how much free space** is left in each tablespace.
- `DBA_FREE_SPACE` is a data dictionary view.
- It sums up the bytes, converting it to **GB** (`/1024/1024/1024`).

Page 13: Add More Space to Tablespace

sql

```
ALTER TABLESPACE users  
ADD DATAFILE 'ora01/oracle/oradata/booktst_users_02.dbf'  
SIZE 500M;
```

Explanation:

- If your tablespace runs out of space, you can add a new **datafile** to it.
- This command adds a **500 MB file** to the existing `users` tablespace.

Page 14: More Operations on Tablespace

Make a Tablespace Read-Only:

sql

```
ALTER TABLESPACE mytspace READ ONLY;
```

- No `INSERT`, `UPDATE`, or `DELETE` allowed.
- Useful for archiving.

Make it Writable Again:

sql

```
ALTER TABLESPACE mytspace READ WRITE;
```



Take a Tablespace Offline:

sql

```
ALTER TABLESPACE mytspace OFFLINE;
```

- Data becomes **inaccessible**.
- Used during maintenance.



Bring it Back Online:

sql

```
ALTER TABLESPACE mytspace ONLINE;
```

- Data becomes **visible again**.



Final Summary of Pages 9–14:

Step	Purpose
Step 1	Create tablespace and assign a datafile
Step 2(a)	Create user with default tablespace
Step 2(b)	Create table and optionally assign different TS
Step 3	Query free space in tablespaces
Add Space	Use <code>ALTER</code> to add more datafiles
Read-Only/Offline Ops	Control access to tablespace (for backup, etc.)

Let me know if you'd like a one-page visual summary or practice SQL exercises based on this!