



Physical Database Design

The Physical Database Design deliverable documents the physical storage and access structures for the application's data. This deliverable transforms the logical database design into a design that can be physically implemented. Use this deliverable to design and document the internal storage structures such as tablespaces, indices, and buffers; any other physical storage structures; and design options for the particular database management system (DBMS).

I. IPT Name:		
II. Deliverable Name: Physical Database Design		Date Completed:
III. Contact Information		
	Name	Channel Unit
IPT Sponsor		
Channel Task Manager		
CIO Task Manager		
Contractor Task Manager		
IV. Task Order Number:		

Description

The Physical Database Design is made up of the following deliverables:

- Database Definition
- Database Space Worksheet
- Relational Index Definition
- Tablespace Definition



Database Definition

Identifies a database which makes up part of the Physical Database Design deliverable. It captures the key aspects of the database, such as the various components: tables, indexes, views, and tablespaces. Optionally, it may include description of the disk configuration, sizings, placement, and segment management strategies. Create this deliverable as an entry point for referencing all the components that belong to this database.

Database Definition

General

Type: Database Definition
Version number: 1.0
Version labels: 1.0
CURRENT

Created: 08/24/98 08:33:52 AM
Modified: 10/15/98 09:36:08 AM
Last modified by: UserX
Created by: UserX

Summary

Name: CUSTSVC
Title (Description): Client X Customer Service Production Database

Keywords:

Details

Type: Relational
DBMS vendor: Oracle

Sub-Components

List of sub-components:

Sub-component	Type of Sub-component
---------------	-----------------------



Customer	Relational Table Definition
CustRep	Relational Table Definition
MSEG	Relational Table Definition
CustIndx	Relational Index Definition
MSEGIndx	Relational Index Definition
PSAPMSEGD	Tablespace Definition
PSAPMSEGI	Tablespace Definition

Additional Information

The following section can be used to provide additional information. It is free text only and will not be stored in the associated property pages.

Disk Array Configuration

The Client X production system utilizes a StorageWorks 800 disk array. There are 17 x 4.3G drives available for use by production. This space is mirrored by 17 additional drives. Two dual HSZ50 controllers will be used. Currently, two single HSZ50 controllers are in place.

The recommended configuration for production is as follows:

- The controllers are configured so all disks can be accessed in the event of any controller failure.
- All disks used by the production system are mirrored.
- Disks that will support database data files are striped:
 - 4 striped sets each containing 3 disks
 - 1 striped set containing 2 disks
- Disks in a striped set reside on separate channels.
- The recommended stripe size for the striped disks is 32K. If it is not possible to configure the disks with a 32K stripe size, the next best size is 64K.
- When the dual controllers are in place, disks are balanced across controllers.
- Three disks will not use striping. These will hold redo and archived log files.
- The 3 disks that are not striped are balanced across controllers so that redo logs can be on separate controllers.

Datafile Sizing and Placement

The datafiles should be sized and placed as outlined in the Database Layout deliverable. Filesystems should be created to support placement of files as outlined.

During the installation process, the datafile sizes should be adjusted. Following the database build, proper creation of the tablespaces and corresponding files must be verified.

Datafile resizing should not be used, due to possible recovery problems involving the control file.



Log Sizing

Redo logs should be sized at 50MB each. The 2 sets of filesystems for redo logs should be placed on separate controllers. The redo logs are created during the database build. The filesystems for the redo logs have been sized to allow increasing redo log size if warranted by future activity.

Rollback Segment Sizing

Three rollback tablespaces in separate stripe sets will help to distribute rollback activity. Create 5 rollback segments in each tablespace. For example, PRS_1, PRS_4, PRS_7, PRS_10, PRS_13 in PSAPROLL1, PRS_2, PRS_5, PRS_8, PRS_11, and PRS_14 in PSAPROLL2, etc. Recommended starting rollback segment parameters are as follows:

Initial	1M	
Next	1M	
Minextents	20	
Maxextents	505	
Optimal		20M

The rollback segment changes can be implemented at a later time following the database build.

Temporary Segment Sizing

Recommended settings for temporary segments are as follows:

Initial	5 x sort_area_size + 1 db block = 10493952
Next	10493956

Sort_area_size should be 2097152.

The temporary tablespace default parameters can be implemented at a later time following the database build.

Large Segment Management

The disk sizing questionnaire available through each hardware vendor and the database sizing tool, when used properly with valid data, can yield useful database sizing estimates. This process can also provide detailed information on expected segment growth. There are no plans to complete this process for Client X. Instead, the system should be monitored closely following conversion so that segments with high growth rates can be managed properly.

Initially, only one table and its corresponding indexes will be moved to a custom tablespace. Following installation, a table reorg should be performed to move MSEG to the tablespace PSAPMSEGD. The corresponding indexes should be moved to PSAPMSEGI. Set the next extent size for MSEG to 250M.



Following conversion and go-live, monitor growth of segments closely. The fastest growing transaction tables should be reviewed as candidates to be moved into separate tablespaces. A general guideline is to try to keep tablespaces under 10G over the long term.

Next Extent Size Adjustment

Default next extent sizes are often too small for productive operation. Active tables and indexes in the development systems provide additional information that can be used to guide adjustments to next extent sizes for production. By monitoring and analyzing extent data from the development systems and by working closely with the application and conversion teams, the database administrator should determine appropriate extent sizes for active objects. The database administrator should then implement the appropriate next extent sizes for the objects in the production system.

Adjusting extent sizes can be a monumental and tedious task. One approach to implementing extent size adjustments is to develop scripts to set sizes based on sizes in development. Sizes for specific objects should be further adjusted based on information from developers.

Extent sizes should be multiples of the database block size (8K). The extent sizes used in a tablespace should be integral multiples or divisors of each other.

Parameter Settings

Platform and system configuration specific adjustments need to be made to the delivered parameter file. The Oracle performance guide should be consulted for tuning. Ongoing monitoring and iterative tuning will be needed in order to achieve and maintain optimal operation of the system.

The following initial parameter recommendations are for Oracle 7. Unless specified, the provided values should be used. If no value is provided, use the Oracle default.

- Start with a shared pool and db buffer cache of 100M each (`shared_pool_size = 104857600`, `db_block_buffers = 12800`).
- Set the `log_buffer = 1048576`.
- Default values should be adequate for managing sorts.
- Uncomment `cursor_space_for_time = true`; set to false if memory becomes an issue.
- Set `log_simultaneous_copies = 4`; this is 2 times the number of processors.
- Use `spin_count = 2000` (default value).
- Leave `_cpu_count` commented out unless otherwise directed by platform specific performance guide.
- Use Oracle defaults for `log_archive_buffer_size (=32)` and `log_archive_buffers (=4)`.
- Check platform specific guide for `readv` parameter. Typically use true value for filesystems.
- Use `db_file_multiblock_read_count = 32`.
- Leave `sequence_cache_entries` and `sequence_cache_hash_buckets` commented out; use Oracle defaults.
- Use rollback segments as outlined in part 1 of this document.
- Use at least 3 controlfiles and ensure that they are placed on separate disks and separate controllers.



Department of Education Student Financial Assistance

- Check platform specific guide for db_writers. Set to 1 if asynchronous I/O is supported for filesystems and activated. Otherwise, set db_writers to the number of controllers.
- Set timed_statistics = true.
- May want to adjust max_dump_file_size.

Ongoing monitoring and tuning will be needed to maintain optimal performance.



Database Space Worksheet

This deliverable describes in detail the assumptions used to calculate the space requirements for a database. Use this deliverable to document the space requirements for a physical database.



Table/Index Description and DASD									
[1] Table Short ID	[2] Index Name	[3] Table Name	[4] Lock Size	[5] UCP	[6] Row/Index Length	[7] Average Row Nbr	[8] Average DASD required	[9] Maximum Row Nbr	[10] Maximum DASD required
CUST		CUSTOMER	Page		72	100,000	8,352	280,000	23,352
	CUST1			U	10		1,568		4,344
INV		INVOICE	Page		120	6,600,000	880,020	13,200,000	1,760,020
	INVOICE1			U	19		168,128		336,232
	INVOICE2			U	70		549,796		1,099,568
VNDR									
		VENDOR	Page	U	20	50	20	100	20
	VENDOR1				10		24		24



Table Space/Column Parameters											
[11] Nbr of VARCHAR Columns	[12] Nbr of Nullable Columns	[13] Index Sub Pages	[14] Avg # Dup Index Entries	[15] Free Space w/i Page (PCTFREE)	[16] Freq of Free Pages (FREEPAGE)	[17] Usable Page Size	[18] Row/Index Length w/ Ovhd	[19] Entries/Page with PCTFREE	[20] Entries/Page w/o PCTFREE	[21] Max Wasted on Pg with PCTFREE	[22] Max Wasted on Pg w/o PCTFREE
0	0	N/A	N/A	5%	0	4,074	80	48	50	234	74
N/A	0	1	1	10%	0	4,050	10	260	289	410	4
0	0	N/A	N/A	5%	0	4,074	128	30	31	234	106
N/A	0	1	1	10%	0	4,050	19	158	176	416	2
N/A	0	1	1	10%	0	4,050	70	49	54	424	54
0	0	N/A	N/A	5%	0	4,074	28	127	127	518	518
N/A	0	1	1	10%	0	4,050	10	260	289	410	4



Average Data/Index Page Calculations									
[23] Number of Pages w/o FREEPAGE	[24] Number of Pages w/ FREEPAGE	[25] Nbr of Index Pages						[26] Index Levels	[27] Total Pages
		Leaf	Level 2	Level 3	Level 4	Level 5	Level 6		
2,086	2,086	0	0	0	0	0	0	0	2,086
387	387	387	2	1	0	0	0	5	390
220,003	220,003	0	0	0	0	0	0	0	220,003
41,775	41,775	41,775	252	2	1	0	0	5	42,030
134,696	134,696	134,696	2,694	54	2	1	0	6	137,447
3	3	0	0	0	0	0	0	0	3
3	3	3	1	0	0	0	0	4	4



Maximum Data/Index Page Calculations									
[28] Number of Pages w/o FREEPAGE	[29] Number of Pages w/ FREEPAGE	[30] Nbr of Index Pages						[31] Index Levels	[32] Total Pages
		Leaf	Level 2	Level 3	Level 4	Level 5	Level 6		
5,836	5,836	0	0	0	0	0	0	0	5,836
1,079	1,079	1,079	4	1	0	0	0	3	1,084
440,003	440,003	0	0	0	0	0	0	0	440,003
83,547	83,547	83,547	504	4	1	0	0	4	84,056
269,390	269,390	269,390	5,388	108	3	1	0	5	274,890
3	3	0	0	0	0	0	0	0	3
3	3	3	1	0	0	0	0	2	4



Extra Calculations										
[33] Average Data DASD	[34] Average Index DASD	[35] Maximum Data DASD	[36] Maximum Index DASD	Nbr of Table Spaces w/ Locking at				[41] Row Length	[42] Index Length	[43] Entries/ Non-Leaf Pages
				[37] Page	[38] Space	[39] Table	[40] Any			
8,352	0	23,352	0	1	0	0	0	72	0	0
0	1,568	0	4,344	0	0	0	0	0	10	281
880,020	0	1,760,020	0	1	0	0	0	120	0	0
0	168,128	0	336,232	0	0	0	0	0	19	166
0	549,796	0	1,099,568	0	0	0	0	0	70	50
20	0	20	0	1	0	0	0	20	0	0
0	24	0	24	0	0	0	0	0	10	281



Relational Index Definition

This deliverable defines a physical index that provides an access path onto a relational table. It identifies the columns that constitute the access path. For all applications using relational databases, use the Relational Index Definition deliverable to describe the characteristics of an index of the table.

Relational Index Definition

General

Type:	Relational Index Definition
Version number:	1.0
Version labels:	1.0 CURRENT

Created:	08/24/98 08:33 AM
Modified:	10/15/98 09:36 AM
Last modified by:	UserX
Created by:	UserX

Summary

Name:	CustIDIndx
Title (Description):	Creates index by Customer ID

Keywords:

Details

Unique?:	Yes
----------	-----

Columns

List of columns:



**Department of Education
Student Financial Assistance**

Data element	Descending?
Cust_ID	No

Additional Information

The following section can be used to provide additional information. It is free text only and will not be stored in the associated property pages.



Tablespace Definition

This deliverable describes the details of the tablespaces defined for this database. Use this deliverable to document the grouping of tables, locking, allocation of freespace, etc.

Tablespace Definition

General

Type: Tablespace Definition
Version number: 1.0
Version labels: 1.0
CURRENT

Created: 08/24/98 08:33 AM
Modified: 10/21/98 07:06 PM
Last modified by: UserX
Created by: UserX

Summary

Name: XYZ
Title (Description): Describes the rationale for the physical database design used.

Keywords:

Tables

List of tables:

Table
TX01_Cust_Type
TX02_Customer
TX03_Order
TX04_Stock_Item
TX05_Supplier

last printed 06/28/00



TX06_Stock_Item_Supplier
TX07_Batch_Control
TX08_System Control

Additional Information

The following section can be used to provide additional information. It is free text only and will not be stored in the associated property pages.

Physical Storage Strategy

In the development database, physical objects will be created by use of the DBMS STOGROUP. Physical location of the data files is then managed outside of the DBMS. This simplifies maintenance for the development DBAs.

In production, physical location of data over the available DASD will be managed and controlled by the IS Data Services Group.

The tables defined in the Logical Database design have been grouped into recovery islands according to the following criteria:

- volatility of data; volatile tables are not grouped with static tables
- functional similarity; tables that mainly support one functional area are grouped together to minimize the number of functions that are affected if problems occur with a particular recovery island, or if a partial recovery of data is required
- size; where possible, the size of a recovery island is minimized so that the elapsed time for backup of the island does not extend the critical path.

Four recovery islands will be implemented:

BCSSA001	-	Customer data: TX01_Cust_Type TX02_Customer
BCCSB001	-	Order data: TX03_Order
BCCSC001	-	Supplier and Stock data: TX04_Stock_Item TX05_Supplier TX06_Stock_Item_Supplier



BCCSD001 - System Data
TX07_Batch_Control
TX08_System Control

All tables are implemented within their own tablespace (other than partitioned tablespaces as described below). This allows for independent control of tablespace storage parameters, such as freespace allocation, for each table.

All tables that are not partitioned will be defined as segmented. This is particularly important for large tables, as the data is managed more easily by the DBMS using this feature.

Data Partitioning Strategy

All tables that at full volume will require more than 50,000 data pages will be partitioned (separate tablespaces created for each partition of the table). Partitioning allows for faster backup, reorganization and recovery of large tables.

For the defined Logical Database Design two tables will be partitioned:

TX02_Customer
TX03_Order

The partitioning index of a partitioned table cannot be updated. Therefore, partitioned indexes are only defined on non-updateable columns (inserts/deletes are allowable). Since the primary keys of all tables are non-updateable, they are chosen as the clustering index by default. There are no exceptions to this rule for the logical data model defined.

Freespace Strategy

The following default Freespace definitions for tablespaces will be used:

Static tables: Freespace 0% PCTFREE 5%

Volatile tables (inserts only) Freespace 5% PCTFREE 5%

in sequential primary key order:

Volatile tables (inserts/deletions) Freespace 10% PCTFREE 20%

in random primary key:

TX08_System_Control is defined with a PCTFREE of 99%, as described in section (8) below.

The freespace definitions should be monitored by the Production DBAs following implementation of the application and the default values modified where appropriate to minimize fragmentation of the database.

Locking Strategy



Department of Education Student Financial Assistance

The application locking approach is as described in the Application Technical Design Report: that is, optimistic locking is used in both batch and online environments using cyclic count numbers. Locking of the datapages during periods of updates will be managed by the DBMS. This is achieved by use of the LOCKSIZE ANY feature in the tablespace definition (with one exception). The DBMS controls the number of locks held in the lock manager by escalating locks from page to tablespace to database as the number of locks held increases. No explicit lock commands should be issued in application code.

The LOCKSIZE of TX08_System_Control is set to PAGE (and a PCTFREE of 99%). Where multiple rows are held on this table, this definition forces each row onto a single data page. Lock escalation to the Tablespace level will not be allowed for this table, as traffic analysis indicates that multiple updates are performed against this table from a large number of areas of the application. Escalation would lead to contention for resources on this table.