General Reference-2.The Data Dictionary

Altibase 7.3

Altibase® Administration

ALTIBASE

Table Of Contents

Preface

About This Manual

This manual explains the concepts, components, and basic use of Altibase

Audience

This manual has been prepared for the following Altibase users:

- Database managers
- Performance managers
- Database users
- Application developers
- Technical support engineers

It is recommended for those reading this manual possess the following background knowledge:

- Basic knowledge in the use of computers, operating systems, and operating system utilities
- Experience in using relational databases and understanding of database concepts
- Computer programming experience
- Experience in database server management, operating system management, or network administration

Organization

This manual has been organized as follows:

- Chapter 1. The Data Dictionary
  This chapter describes the specification of the Altibase data dictionary. The data dictionary of Altibase consists of meta tables, in which information about objects is stored, and the process tables, in which information about processes is stroed.
- Chapter 2. Sample Schema
  This chapter describes the example table information, ER diagram, and sample data.

Documentation Conventions
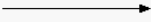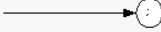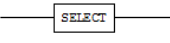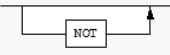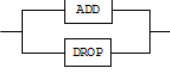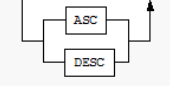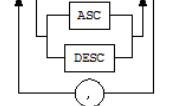
This section describes the conventions used in this manual. Understanding these conventions will make it easier to find information in this manual and other manuals in the series.

There are two sets of conventions:

- Syntax diagrams
- Sample code conventions

Syntax diagrams

This manual describes command syntax using diagrams composed of the following elements:

Elements	Meaning
Reserved Word	Indicates the start of a command. If a syntactic element starts with an arrow, it is not a complete command.
→	Indicates that the command continues to the next line. if a syntactic element ends with this symbol, it is not a complete command.
→	Indicates that the command continues from the previous line. If a syntactic element starts with this symbol, it is not a complete command.
→;	Indicates the end of a statement.
SELECT	Indicates a mandatory element.
NOT	Indicates an optional element.
ADD / DROP	Indicates a mandatory element comprised of options. One, and only one, option must be specified.
ASC / DESC	Indicates an optional element comprised of options.
ASC / DESC	Indicates an optional element in which multiple elements may be specified. A comma must precede all but the first element.

Sample Code Conventions

The code examples explain SQL, stored procedures, iSQL, and other command line syntax.

The following describes the conventions used in the code examples:

Rules	Meaning	Example
[ ]	Indicates an optional item	VARCHAR [(*size*)] [[FIXED |] VARIABLE]
{ }	Indicates a mandatory field for which one or more items must be selected.	{ ENABLE | DISABLE | COMPILE }
|	A delimiter between optional or mandatory arguments.	{ ENABLE | DISABLE | COMPILE } [ENABLE | DISABLE | COMPILE]

Rules	Meaning	Example
. . .	Indicates that the previous argument is repeated, or that sample code has been omitted.	SQL> SELECT ename FROM employee; ENAME ---------------------- SWNO HJNO HSCHOI . . . 20 rows selected.
Other Symbols	Symbols other than those shown above are part of the actual code.	EXEC :p1 := 1; acc NUMBER(11,2)
Italics	Statement elements in italics indicate variables and special values specified by the user.	SELECT * FROM *table_name*; CONNECT *userID*/*password*;
Lower case words	Indicate program elements set by the user, such as table names, column names, file names, etc.	SELECT ename FROM employee;
Upper case words	Keywords and all elements provided by the system appear in upper case.	DESC SYSTEM.*SYS_INDICES*;

Related Documents

For more detailed information, please refer to the following documents:

- Installation Guide
- Getting Started Guide
- Administrator's Manual
- Replication Manual

Altibase Welcomes Your Comments and Feedback

Please let us know what you like or dislike about our manuals. To help us with better future versions of our manuals, please tell us if there are any corrections or classifications that you would find useful.

Include the following information:

- The name and version of the manual that you are using
- Any comments about the manual
- Your name, address, and phone number

If you need immediate assistance regarding any errors, omissions, and other technical issues, please contact Altibase's Support Portal.

Thank you. We always welcome your feedback and suggestions.

1. The Data Dictionary

The data dictionary of Altibase consists of meta tables, in which information about objects is stored, and process tables, in which information about processes is stored. Process tables comprise fixed tables and performance views.

This chapter describes the Altibase data dictionary, which is the basis of all database objects and all Altibase system information.

Meta Tables

Meta tables are system-defined tables that contain all information about database objects.

This section describes the types of meta tables and their structure, and explains how to read and update the information in meta tables.

Structure and Function

Meta tables are defined by the system for the purpose of managing database objects. They use the same data types and store records in the same way as user-defined tables. When Altibase starts up, it loads information about database objects. When DDL statements are executed, meta tables are used to read, store, and update this information. The owner of meta tables is the system user (user name: SYSTEM_), so normal users have limited access to meta tables.

Retrieving Information from Meta Tables

When a database object is created, deleted or modified using a DDL statement, the system creates, deletes, or updates records in one or more meta tables.

After a DDL statement is executed, the resultant changes to database objects can be confirmed by checking meta tables. This is accomplished using a SELECT statement, just as with a regular database table.

Modifying Data in Meta Tables

It is possible to use DML statements to explicitly make changes to the data in meta tables. However, the system-defined system user (SYSTEM_) can only make such changes to meta tables. Additionally, when the information in meta tables is changed, the system may become impossible to start, information about database objects may be lost, or the system may be critically damaged. Therefore, users must avoid making changes to meta tables whenever possible. When it is inevitable that a user must change meta table information, it is imperative that the database first be backed up, and it must be understood that the user is completely responsible for any damage resulting from making direct changes to meta table information.

Modifying Meta Table Schema

The meta table schema may be modified when a new kind of DDL statement is introduced, or when the functionality of an existing statement is changed. Depending on the characteristics of the changes to meta table schema, one of two cases may arise: either the database might need to be migrated, or the meta table schema will simply be automatically modified when Altibase is restarted. This should

be kept in mind when upgrading Altibase to a newer version.

The types of Meta Tables

This table shows the list of meta tables. Their names start with SYS_.

Meta Table Name	Description
SYS_AUDIT_	This table stores information about the operation status of the audit.
SYS_AUDIT_OPTS_	This view stores auditing conditions. SYS_AUDIT_ALL_OPTS_ is the base table of this view.
SYS_COLUMNS_	This table contains information about columns.
SYS_COMMENTS_	This table contains information about explanatory comments.
SYS_COMPRESSION_TABLES_	This table contains information about compressed columns.
SYS_CONSTRAINTS_	This table contains information about constraints.
SYS_CONSTRAINT_COLUMNS_	This table contains information about columns having constraints.
SYS_CONSTRAINT_RELATED_	This table contains information about the stored functions referenced by the constraints.
SYS_DATABASE_	This table contains information about the name and version of the database.
SYS_DATABASE_LINKS_	This table contains information about the database links.
SYS_DIRECTORIES_	This table contains information about directories used by stored procedures for managing files.
SYS_DN_USERS_	This table is reserved for future use.
SYS_DUMMY_	This table is for internal use only.
SYS_ENCRYPTED_COLUMNS_	This table contains additional security information for individual columns.
SYS_GRANT_OBJECT_	This table contains information about object privileges.
SYS_GRANT_SYSTEM_	This table contains information about system privileges.
SYS_INDEX_COLUMNS_	This table contains information about index key columns.
SYS_INDEX_PARTITIONS_	This table contains information about index partitions.

Meta Table Name	Description
SYS_INDEX_RELATED_	This table contains information about the stored functions on which the function-based indexes are based.
SYS_INDICES_	This table contains information about indexes.
SYS_JOBS_	This table contains information about jobs.
SYS_LIBRARIES_	This table contains information about external library objects.
SYS_LOBS_	This table contains information about LOB columns.
SYS_MATERIALIZED_VIEWS_	This table contains information about materialized view.
SYS_PACKAGES_	This table contains information about packages.
SYS_PACKAGE_PARAS_	This table contains information about subprogram(stored procedures and stored functions) parameters contained in packages.
SYS_PACKAGE_PARSE_	This table contains information about statement texts of user-defined packages.
SYS_PACKAGE_RELATED_	This table contains information about tables, sequences, stored procedures, stored functions, or views accessed by stored procedures and stored functions inside packages.
SYS_PART_INDICES_	This table contains information about partitioned indexes.
SYS_PART_KEY_COLUMNS_	This table contains information about partitioning keys.
SYS_PART_LOBS_	This table contains information about LOB columns for respective partitions.
SYS_PART_TABLES_	This table contains information about partitioned tables.
SYS_PASSWORD_HISTORY_	This table contains information about alterations made to user passwords that have been assigned a password policy.
SYS_PASSWORD_LIMITS_	This meta table contains specified password management policies at user creation and account status quo.
SYS_PRIVILEGES_	This table contains information about privileges.
SYS_PROCEDURES_	This table contains information about stored procedures and functions.

Meta Table Name	Description
SYS_PROC_PARAS_	This table contains information about the parameters for stored procedures and functions.
SYS_PROC_PARSE_	This table contains the actual text of stored procedures and stored functions.
SYS_PROC_RELATED_	This table contains information about tables accessed by stored procedures and functions.
SYS_RECYCLEBIN_	The table contains information about tables in the recycle bin.
SYS_REPLICATIONS_	This table contains general information about replication.
SYS_REPL_HOSTS_	This table contains information about replication hosts.
SYS_REPL_ITEMS_	This table contains information about tables to be replicated
SYS_REPL_OFFLINE_DIR_	This table contains information about the log directory related to the replication offline option.
SYS_REPL_OLD_CHECKS_	This table contains information about replication target columns that is being replicated by replication sender thread and has CHECK constraints.
SYS_REPL_OLD_CHECK_COLUMNS_	This meta table contains information about CHECK constraints on replication target column that replication sender thread is currently processing.
SYS_REPL_OLD_COLUMNS_	This table contains information about columns replicated by the replication sender thread.
SYS_REPL_OLD_INDEX_COLUMNS_	This table contains information about index columns replicated by the replication sender thread.
SYS_REPL_OLD_INDICES_	This table contains information about indexes replicated by the replication sender thread.
SYS_REPL_OLD_ITEMS_	This table contains information about the tables replicated by the replication sender thread.
SYS_REPL_TABLE_OID_IN_USE_	This table contains information about TABLE OID of tables included in DDL log but not yet replicated.
SYS_REPL_RECOVERY_INFOS_	This table contains information about logs used by replication for recovery of a remote server.
SYS_SECURITY_	This table contains information about the state of the security module.

Meta Table Name	Description
SYS_SYNONYMS_	This table contains information about synonyms.
SYS_TABLES_	This table contains information about all kinds of tables.
SYS_TABLE_PARTITIONS_	This table contains information about table partitions.
SYS_TABLE_SIZE_	This table contains information about the actual size of disk and memory tables in the system.
SYS_TBS_USERS_	This table contains information about users' access to user-defined tablespaces.
SYS_TRIGGERS_	This table contains information about triggers.
SYS_TRIGGER_DML_TABLES_	This table contains information about tables accessed by triggers.
SYS_TRIGGER_STRINGS_	This table contains the actual text of trigger commands.
SYS_TRIGGER_UPDATE_COLUMNS_	This table contains information about columns that cause triggers to fire whenever their contents are changed.
SYS_USERS_	This table contains information about users.
DBA_USERS_	The DBA_USERS is a meta table which stores the user information. Only SYS can make an inquiry.
SYS_USER_ROLES_	This table stores information about the roles granted to the user.
SYS_VIEWS_	This table contains information about views.
SYS_VIEW_PARSE_	This table contains the actual text of statements used to create views.
SYS_VIEW_RELATED_	This table contains information about objects accessed by views.
SYS_XA_HEURISTIC_TRANS_	This table contains information about global transactions.
SYS_GEOMETRIES_	This table contains information about tables that have GEOMETRY columns.
SYS_GEOMETRY_COLUMNS_	This table contains information about GEOMETRY columns; The synonym of this meta table is GEOMETRY_COLUMNS_.
USER_SRS_	This table contains information about SRS(Spatial Reference System); The synonym of this meta table is SPATIAL_REF_SYS

Unsupported Meta Tables

Altibase provides the following GIS-related meta tables. Their names begin with STO_. They are currently unsupported.

```
STO_COLUMNS_
STO_DATUMS_
STO_ELLIPSOIDS_
STO_GEOCCS_
STO_GEOGCS_
STO_PRIMEMS_
STO_PROJCS_
STO_PROJECTIONS_
STO_SRS_
STO_USER_COLUMNS_
```

SYS_AUDIT_

This meta table stores information about the operation status of the audit.

Column name	Type	Description
IS_STARTED	INTEGER	Whether or not auditing is being executed
START_TIME	DATE	The time at which auditing started
STOP_TIME	DATE	The time at which auditing stopped
RELOAD_TIME	DATE	The time at which the auditing conditions were applied to the server

Column Information

IS_STARTED

Indicates whether or not auditing is currently being performed.

0: Auditing is currently not being performed.

1: Auditing is currently being performed.

START_TIME

Indicates the time at which auditing started.

STOP_TIME

Indicates the time at which auditing stopped.

RELOAD_TIME

Indicates the time at which altered auditing conditions were applied to the Altibase server. The value of this column is updated for the occasions below:

- When the DBA has started auditing, using the ALTER SYSTEM START AUDIT statement.
- When the DBA has applied altered auditing conditions to auditing, using the ALTER SYSTEM RELOAD AUDIT statement.

SYS_AUDIT_OPTS_

This meta view stores auditing conditions. The base table of this view is the SYS_AUDIT_ALL_OPTS_ meta table.

Column name	Type	Description
USER_NAME	VARCHAR(128)	The user name
OBJECT_NAME	VARCHAR(128)	The object name
OBJECT_TYPE	VARCHAR(40)	The object type
SELECT_OP	CHAR(3)	The units in which logs are written for each operation statement
INSERT_OP	CHAR(3)	
UPDATE_OP	CHAR(3)	
DELETE_OP	CHAR(3)	
MOVE_OP	CHAR(3)	
MERGE_OP	CHAR(3)	
ENQUEUE_OP	CHAR(3)	
DEQUEUE_OP	CHAR(3)	
LOCK_TABLE_OP	CHAR(3)	
EXECUTE_OP	CHAR(3)	
COMMIT_OP	CHAR(3)	
ROLLBACK_OP	CHAR(3)	
SAVEPOINT_OP	CHAR(3)	
CONNECT_OP	CHAR(3)	
DISCONNECT_OP	CHAR(3)	
ALTER_SESSION_OP	CHAR(3)	
ALTER_SYSTEM_OP	CHAR(3)	

DDL_OP	CHAR(3)	

Column Information

USER_NAME

This is the user name of the owner of the auditing target object.

OBJECT_NAME

This is the name of the auditing target object.

OBJECT_TYPE

This is the type of the target object, which is one of the following:

- TABLE
- VIEW
- QUEUE
- SEQUENCE
- PROCEDURE
- FUNCTION

XXX_OP

This is the units for logs of operation statements. Before '/' is the unit for logs of successful executions, and after is the unit for logs of failed executions.

The units for logs are as below:

- -: Logs are not written.
- S: Logs are written in the unit of sessions.
- A: Logs are written in the unit of accesses.
- T: Logs are written regardless the unit of session or accesses.

The following examples show values of the SYS_AUDIT_OPTS_ view after auditing conditions are enabled.

```
iSQL> AUDIT insert, select, update, delete on friends BY SESSION WHENEVER
SUCCESSFUL;
Audit success.

iSQL> AUDIT insert, select, update, delete on friends BY ACCESS WHENEVER NOT
SUCCESSFUL;
Audit success.

USER_NAME : SYS
OBJECT_NAME : FRIENDS
OBJECT_TYPE : TABLE
SELECT_OP : S/A
```

```
INSERT_OP : S/A
UPDATE_OP : S/A
DELETE_OP : S/A
MOVE_OP : -/-
MERGE_OP : -/-
ENQUEUE_OP : -/-
DEQUEUE_OP : -/-
LOCK_TABLE_OP : -/-
EXECUTE_OP : -/-
COMMIT_OP : -/-
ROLLBACK_OP : -/-
SAVEPOINT_OP : -/-
CONNECT_OP : -/-
DISCONNECT_OP : -/-
ALTER_SESSION_OP : -/-
ALTER_SYSTEM_OP : -/-
DDL_OP : -/-

iSQL> AUDIT DDL BY sys WHENEVER NOT SUCCESSFUL;
Audit success.


USER_NAME : SYS
OBJECT_NAME : ALL
OBJECT_TYPE :
SELECT_OP : -/-
INSERT_OP : -/-
UPDATE_OP : -/-
DELETE_OP : -/-
MOVE_OP : -/-
MERGE_OP : -/-
ENQUEUE_OP : -/-
DEQUEUE_OP : -/-
LOCK_TABLE_OP : -/-
EXECUTE_OP : -/-
COMMIT_OP : -/-
ROLLBACK_OP : -/-
SAVEPOINT_OP : -/-
CONNECT_OP : -/-
DISCONNECT_OP : -/-
ALTER_SESSION_OP : -/-
ALTER_SYSTEM_OP : -/-
DDL_OP : -/T
```

SYS_COLUMNS_

Information about columns defined in all tables, virtual columns in all views, and virtual columns in all sequences is stored in this meta table.

Column name	Type	Description

Column name	Type	Description
COLUMN_ID	INTEGER	The column identifier
DATA_TYPE	INTEGER	The data type
LANG_ID	INTEGER	The language identifier
OFFSET	BIGINT	The offset of the column within the record
SIZE	BIGINT	The physical length of the column within the record
USER_ID	INTEGER	The user identifier
TABLE_ID	INTEGER	The table identifier
PRECISION	INTEGER	The specified precision of the column
SCALE	INTEGER	The specified scale of the column
COLUMN_ORDER	INTEGER	The position of the column in the table
COLUMN_NAME	VARCHAR(128)	The name of the column
IS_NULLABLE	CHAR(1)	Whether NULL is permitted. T: can be NULL F: cannot be NULL
DEFAULT_VAL	VARCHAR(4000)	The default value or expression
STORE_TYPE	CHAR(1)	The column storage type V: variable type F: fixed type L: LOB column
IN_ROW_SIZE	INTEGER	The length of data that can be saved in a fixed area when data are saved in a variable-length column in a memory table
REPL_CONDITION	INTEGER	Deprecated
IS_HIDDEN	CHAR(1)	Whether the column is hidden or not T: hidden column F: public column
IS_KEY_PRESERVED	CHAR(1)	Whether or not the column is modifiable T: Modifiable F: Unmodifiable

Column Information

COLUMN_ID

This is the column identifier, which is assigned automatically by the system sequence.

DATA_TYPE

This is the data type identifier. The identifiers for each data type are as follows:

Data Type	Value
CHAR	1
VARCHAR	12
NCHAR	-8
NVARCHAR	-9
NUMERIC	2
DECIMAL	2
FLOAT	6
NUMBER	6
DOUBLE	8
REAL	7
BIGINT	-5
INTEGER	4
SMALLINT	5
DATE	9
BLOB	30
CLOB	40
BYTE	20001
NIBBLE	20002
BIT	-7
VARBIT	-100
GEOMETRY	10003

For more information about data types, please refer to Chapter1: Data Types.

LANG_ID

A column that contains the language properties for character data types (CHAR, VARCHAR).

OFFSET

This indicates the physical starting point of a column within a record. The offset and size of a column are used to calculate the physical storage size of a record.

SIZE

This is the physical storage size of the column in a record, calculated by the system based on the column type, user-defined precision, etc.

USER_ID

This corresponds to a USER_ID value in the SYS_USERS_ meta table, and identifies the owner of the table to which the column belongs.

TABLE_ID

This corresponds to a TABLE_ID value in the SYS_TABLES_ meta table, and identifies the table to which the column belongs.

PRECISION

This is the precision of the data type, and is either defined by the user or corresponds to the default value for the system. In the case of a character data type, it corresponds to the length of the character data type set by the user.

SCALE

This is the scale of the data type, and is either defined by the user or corresponds to the default value for the system. This value is not used with some data types.

COLUMN_ORDER

This is the order in which columns appear in a table.

The order in which the columns are stated in a CREATE TABLE statement determines the order in which they are created, and thus their position in the table. If a column is added using an ALTER TABLE statement, the newly created column will be the last column in the table.

COLUMN_NAME

This is the name specified when a user creates a table or adds a column to the table.

IS_NULLABLE

Indicates whether NULL is allowed in the column.

When creating a column, the user can explicitly specify whether to allow NULL for the column. If not specified, NULL is allowed by default.

DEFAULT_VAL

The default value the user specified in the column is displayed.

If the column is a hidden column added automatically due to the creation of a function-based index, the formula used to create the function-based index is stored.

STORE_TYPE

When physically storing a column, it can either be written as part of a record, or it can be saved on another page, in which case only the location of the data is stored in the record.

If the physical storage size of a column is too big, or if the size of the column varies frequently for individual records, the column can be stored on another page by using the VARIABLE option when defining the column. This option is generally used for VARCHAR types where the character strings in a column are long.

This column indicates whether the VARIABLE option is used.

IN_ROW_SIZE

This is the default IN_ROW_SIZE when data are stored in variable-length columns in memory tables. When data are inserted into a variable-length column, if the length of the data is equal to or smaller than the value specified by IN_ROW_SIZE, the data are stored in the fixed space, whereas if the data are longer than this value, they are stored in a variable space. For disk tables, this value is always 0.

For more information about variable-length columns and the IN ROW clause, please refer to Chapter1: Data Types.

IS_HIDDEN

This indicates whether the column has hidden properties or not. On the creation of function-based indexes, columns with hidden properties are automatically added to the table. One of the following two values is displayed in this column:

- T: Hidden column
- F: Public column

IS_KEY_PRESERVED

This indicates whether the column of the join view is modifiable with DML statements(INSERT, UPDATE, DELETE). For columns of regular tables, this value is specified as 'T'; for views, this value is specified as 'T' for modifiable columns, and 'F' for unmodifiable columns.

- T: Modifiable columns
- F: Unmodifiable columns

Reference Tables

```
SYS_USERS_
SYS_TABLES_
SYS_GEOMETRIES_
```

SYS_COMMENTS_

This meta table is for storing comments such as descriptions of user-defined tables, views and associated columns.

Column name	Type	Description
USER_NAME	VARCHAR(128)	The name of the user
TABLE_NAME	VARCHAR(128)	The name of the table
COLUMN_NAME	VARCHAR(128)	The name of the column
COMMENTS	VARCHAR(4000)	The actual comment

Column Information

USER_NAME

This is the name of the table owner. Its value corresponds to one of the USER_NAME values in the SYS_USERS_ meta table.

TABLE_NAME

This is the name of the table (or view). Its value is the same as one of the TABLE_NAME values appearing in SYS_TABLES_.

COLUMN_NAME

This is the name of a column in the table (or view). Its value is equal to a COLUMN_NAME value in the SYS_COLUMNS_ meta table.

However, if the comment pertains to an entire table (or view), the value for COLUMN_NAME will be NULL.

COMMENTS

This is the actual comment written by the user.

Reference Tables

```
SYS_USERS_
SYS_TABLES_
SYS_COLUMNS_
```

SYS_COMPRESSION_TABLES_

This meta table stores information about compressed columns.

Column name	Type	Description

Column name	Type	Description
TABLE_ID	INTEGER	The identifier of the table with the compressed column
COLUMN_ID	INTEGER	The identifier of the compressed column
DIC_TABLE_ID	INTEGER	The identifier of the dictionary table in which data of the compressed column is actually stored
MAXROWS	BIGINT	The maximum number of rows that can be inserted in the table where data of the compressed column is stored (0: unlimited)

Column Information

TABLE_ID

This is the identifier of the table with the compressed column. This value corresponds to one TABLE_ID value of the SYS_TABLES_ meta table.

COLUMN_ID

This is the identifier of the compressed column. This value corresponds to one COLUMN_ID value of the SYS_COLUMNS_ meta table.

DIC_TABLE_ID

This is the identifier of the dictionary table in which data of the compressed column is actually stored.

MAXROWS

This is the maximum number of rows that can be inserted to the dictionary table where data of the compressed column is actually stored.

SYS_CONSTRAINTS_

This meta table contains information about table constraints.

Column name	Type	Description
USER_ID	INTEGER	The user identifier
TABLE_ID	INTEGER	The table identifier
CONSTRAINT_ID	INTEGER	The constraint identifier
CONSTRAINT_NAME	VARCHAR(128)	The name of the constraint
CONSTRAINT_TYPE	INTEGER	The type of the constraint
INDEX_ID	INTEGER	The identifier of the index used by the constraint
COLUMN_CNT	INTEGER	The number of columns that are associated with the constraint

Column name	Type	Description
REFERENCED_TABLE_ID	INTEGER	The identifier of a table referenced in a FOREIGN KEY constraint
REFERENCED_INDEX_ID	INTEGER	The identifier of an index referenced in a FOREIGN KEY constraint
DELETE_RULE	INTEGER	Whether to perform cascade delete for a FOREIGN KEY constraint 0: Do not perform cascade delete 1: perform cascade delete 2: SET NULL, columns with dependent foreign key values are modified to NULL.
CHECK_CONDITION	VARCHAR(4000)	The character string condition of the CHECK constraint
VALIDATED	CHAR(1)	Whether all data conform to the constraint

Column Information

USER_ID

This is the user identifier, and corresponds to a USER_ID in the SYS_USERS_ meta table.

TABLE_ID

This is the identifier for the table associated with the constraint, and will correspond to a TABLE_ID value in the SYS_TABLES_ meta table.

CONSTRAINT_ID

This is a constraint identifier. It is automatically assigned by the system sequence.

CONSTRAINT_NAME

This is the name of the constraint.

CONSTRAINT_TYPE

This indicates the type of the constraint. The possible types are as follows:

- 0: FOREIGN KEY
- 1: NOT NULL
- 2: UNIQUE
- 3: PRIMARY KEY
- 5: TIMESTAMP
- 6: LOCAL UNIQUE
- 7: CHECK

For additional information about each type of constraint, please refer to the description of column constraints in the explanation of the CREATE TABLE statement in the *SQL Reference.*

INDEX_ID

If an index must be created in order to define constraints such as UNIQUE or PRIMARY KEY constraints, the system creates an index internally. This is the identifier of that index, and will correspond to an INDEX_ID in the SYS_INDICES_ meta table.

COLUMN_CNT

This is the number of columns associated with the constraint. For example, for a constraint such as UNIQUE (i1, i2, i3), this value would be 3.

REFERENCED_TABLE_ID

This is the identifier of a table referenced in a FOREIGN KEY constraint (not the table for which the constraint is defined). This identifier will correspond to a TABLE_ID value in the SYS_TABLES_ meta table.

REFERENCED_INDEX_ID

This indicates a UNIQUE or PRIMARY KEY constraint that must exist in a table referenced by a FOREIGN KEY constraint. The identifier of this constraint will be the same as a CONSTRAINT_ID value in the SYS_CONSTRAINTS_ meta table.

CHECK_CONDITION

This displays the Integrity Rule defined by the user at CHECK constraint specification.

VALIDATED

This indicates whether all data conform to the constraint.

Reference Tables

```
SYS_USERS_
SYS_TABLES_
SYS_INDICES_
```

SYS_CONSTRAINT_COLUMNS_

This meta table contains information about columns related to all constraints defined in user tables.

Column name	Type	Description
USER_ID	INTEGER	The user identifier
TABLE_ID	INTEGER	The table identifier
CONSTRAINT_ID	INTEGER	The constraint identifier
CONSTRAINT_COL_ORDER	INTEGER	The position of the column in the constraint

Column name	Type	Description
COLUMN_ID	INTEGER	The column Identifier

Column Information

USER_ID

This is the user identifier, and corresponds to a USER_ID in the SYS_USERS_ meta table.

TABLE_ID

This is the identifier of the table in which the constraint is defined, and corresponds to a TABLE_ID value in the SYS_TABLES_ meta table.

CONSTRAINT_ID

This is the identifier of the constraint, and corresponds to a CONSTRAINT_ID value in the SYS_CONSTRAINTS_ meta table.

CONSTRAINT_COL_ORDER

This is the position of the column within the constraint. For example, when the constraint UNIQUE (i1,i2,i3) is created, three records are inserted into the SYS_CONSTRAINT_COLUMNS_ meta table. The position of column i1 is 1, column i2 is 2, and column i3 is 3.

COLUMN_ID

This is the identifier of the column for which the constraint is defined, and corresponds to a COLUMN_ID value in the SYS_COLUMNS_ meta table.

Reference Tables

```
SYS_USERS_
SYS_TABLES_
SYS_CONSTRAINTS_
SYS_COLUMNS_
```

SYS_CONSTRAINT_RELATED_

This meta table contains information about the stored functions referenced by the constraints.

Column name	Type	Description
USER_ID	INTEGER	The user identifier
TABLE_ID	INTEGER	The table identifier
CONSTRAINT_ID	INTEGER	The constraint identifier
RELATED_USER_ID	INTEGER	The identifier of the owner of the stored function referenced by the constraint

Column name	Type	Description
RELATED_PROC_NAME	VARCHAR(128)	The name of the stored function referenced by the constraint

Column Information

USER_ID

This is the identifier of the owner of the constraint, and is identical to one USER_ID value in the SYS_USERS_ meta table.

TABLE_ID

This is the identifier of the table which defines the constraint, and is identical to one TABLE_ID value in the SYS_TABLES_ meta table.

CONSTRAINT_ID

This is the identifier of the constraint, and is identical to one CONSTRAINT_ID value in the SYS_CONSTRAINTS_ meta table.

RELATED_USER_ID

This is the identifier of the owner of the stored function referenced by the constraint, and is identical to one USER_ID value in the SYS_USERS_ meta table.

RELATED_PROC_NAME

This is the name of the stored function referenced by the constraint, and is identical to one PROC_NAME value in the SYS_PROCEDURES_ meta table.

Reference Tables

```
SYS_USERS_
SYS_TABLES_
SYS_CONSTRAINTS_
SYS_PROCEDURES_
```

SYS_DATABASE_

This is the table that contains the database name and meta table version information.

Column name	Type	Description
DB_NAME	VARCHAR(40)	The database name
OWNER_DN	VARCHAR(2048)	Reserved for future use
META_MAJOR_VER	INTEGER	The database meta table version (Main)
META_MINOR_VER	INTEGER	The database meta table version (Sub)

Column name	Type	Description
META_PATCH_VER	INTEGER	The database meta table version (Patch)

Column Information

DB_NAME

The database name specified when creating the database is saved.

META_MAJOR_VER

This value increases when a meta table is modified, added or removed. If the database version and the corresponding binary version of Altibase do not match, the database must be migrated.

META_MINOR_VER

This value increases when the contents of one or more meta tables is modified. If the version of the database does not correspond to the current version of Altibase, the system internally compares this value and automatically upgrades the meta tables to the newer version.

META_PATCH_VER

This indicates the meta table patch version.

SYS_DATABASE_LINKS_

This meta table is for storing Database Link information.

Column name	Type	Description
USER_ID	INTEGER	The user identifier
LINK_ID	INTEGER	The Database Link identifier
LINK_OID	BIGINT	The Database Link object identifier
LINK_NAME	VARCHAR(40)	The Database Link name
USER_MODE	INTEGER	The access method to the remote server
REMOTE_USER_ID	VARCHAR(128)	The user account for a remote database
REMOTE_USER_PWD	BYTE(40)	The user password for a remote database
LINK_TYPE	INTEGER	Indicates whether it is a Heterogeneous Link or a Homogeneous Link.
TARGET_NAME	VARCHAR(40)	The name of the remote server which the database link object is to access
CREATED	DATE	The date and time at which the database link object is created

Column name	Type	Description
LAST_DDL_TIME	DATE	The time at which the database link object was most recently changed using a DDL statement

Column Information

USER_ID

This is the identifier of the user who owns the Database Link object.

LINK_ID

This is the Database Link identifier.

LINK_OID

This is the Database Link object identifier.

LINK_NAME

This is the name of the Database Link object, which is specified by the user when the Database Link object is created.

USER_MODE

This indicates the mode in which a remote server is accessed.

- 0: DEDICATE USER MODE
- 1: CURRENT USER MODE (reserved for future use)

REMOTE_USER_ID

This indicates a user account on a remote server, to be used when accessing a remote database server.

REMOTE_USER_PWD

This is the password for the user account on the remote server, to be used when accessing a remote database server. The password is encrypted using an encryption algorithm before it is stored.

LINK_TYPE

This indicates whether a heterogeneous link or a homogeneous link.

TARGET_NAME

This indicates the name of the remote server that the database link object will access.

CREATED

This indicates the date when the database link object was created.

LAST_DDL_TIME

This indicates the last time a DDL change occurred to a database link object.

SYS_DIRECTORIES_

This table contains information about directories that are used when files are managed using stored procedures

Column name	Type	Description
DIRECTORY_ID	BIGINT	The directory identifier
USER_ID	INTEGER	The user identifier
DIRECTORY_NAME	VARCHAR(128)	The directory name
DIRECTORY_PATH	VARCHAR(4000)	The absolute path of the directory on the system
CREATED	DATE	The time at which the directory was created
LAST_DDL_TIME	DATE	The most recent time at which a DDL task was used to change the directory object

Column Information

DIRECTORY_ID

This is a directory identifier. It is a unique value within the system.

USER_ID

This is the user identifier of the owner of the directory.

DIRECTORY_NAME

This is the name of the directory. It is a unique value within the system.

DIRECTORY_PATH

This is the absolute path where the directory is located. This value is explicitly set by the user when executing a CREATE DIRECTORY statement.

LAST_DDL_TIME

This is the most recent time at which a DDL task was used to change the directory object.

SYS_ENCRYPTED_COLUMNS_

This is the meta table for managing additional security information based on the security settings for individual columns.

Column name	Type	Description

Column name	Type	Description
USER_ID	INTEGER	The identifier of the owner of the table to which the column belongs
TABLE_ID	INTEGER	The identifier of the table to which the column belongs
COLUMN_ID	INTEGER	The identifier of the encrypted column
ENCRYPT_PRECISION	INTEGER	The precision of the column encryption
POLICY_NAME	VARCHAR(16)	The name of the encryption policy
POLICY_CODE	VARCHAR(128)	The verification code of the encryption policy

SYS_GRANT_OBJECT_

This contains information about object privileges granted to a user.

Column name	Type	Description
GRANTOR_ID	INTEGER	The identifier of the user who granted the privileges
GRANTEE_ID	INTEGER	The identifier of the user to whom the privileges were granted
PRIV_ID	INTEGER	The privilege identifier
USER_ID	INTEGER	The identifier of the owner of the object
OBJ_ID	BIGINT	The identifier of the object
OBJ_TYPE	VARCHAR(1)	The type of object
WITH_GRANT_OPTION	INTEGER	Indicates whether the WITH_GRANT_OPTION is used when object access privileges are granted 0: Not used 1: Used

Column Information

GRANTOR_ID

This is the identifier of the user who granted the privilege, and corresponds to a USER_ID in the SYS_USERS_ meta table.

GRANTEE_ID

This is the identifier of the user to whom the privilege has been granted, and corresponds to a USER_ID in the SYS_USERS_ meta table. If an object privilege is granted to PUBLIC, however, the USER_ID value "0"(which does not exist in the SYS_USERS_ meta table) is displayed in this column.

PRIV_ID

This is the identifier of the privilege. It corresponds to a PRIV_ID in the SYS_PRIVILEGES_ meta table.

USER_ID

This is the user ID of the owner of the object for which the privilege has been granted. This value will correspond to a USER_ID in the SYS_USERS_ meta table.

OBJ_ID

This is the ID of the object for which the privilege has been granted. It corresponds with one, and only one, target object ID saved in the appropriate meta table.

If the target object is a table, view or sequence, it is mapped to a TABLE_ID in the SYS_TABLES_ meta table, whereas if it is a stored procedure or stored function, it is mapped to a PROC_OID in the SYS_PROCEDURES_ meta table.

OBJ_TYPE

This is the type of the object related to the privilege.

- A: Stored package
- D: Directory
- T: Table or View
- S: Sequence
- P: Stored procedure or function
- Y: Library

WITH_GRANT_OPTION

The WITH_GRANT_OPTION indicates whether the user to whom the privilege was granted is permitted to grant the privilege to other users.

Reference Tables

```
SYS_USERS_
SYS_PRIVILEGES_
SYS_TABLES_
SYS_PROCEDURES_
```

SYS_GRANT_SYSTEM_

This contains information about system privileges granted to users

Column name	Type	Description
GRANTOR_ID	INTEGER	The identifier of the user who granted the privilege
GRANTEE_ID	INTEGER	The identifier of the user to whom the privilege was granted

Column name	Type	Description
PRIV_ID	INTEGER	The identifier of the privilege

Column Information

GRANTOR_ID

This is the identifier of the user who granted the privilege, and corresponds to a USER_ID in the SYS_USERS_ meta table.

GRANTEE_ID

This is the identifier of the user to whom the privilege was granted, and corresponds to a USER_ID in the SYS_USERS_ meta table.

PRIV_ID

This is the identifier of the privilege, and corresponds to a PRIV_ID found in the SYS_PRIVILEGES_ meta table.

Reference Tables

```
SYS_USERS_
SYS_PRIVILEGES_
```

SYS_INDEX_COLUMNS_

This is the meta table that contains information about all columns associated with indexes defined for all tables.

Column name	Type	Description
USER_ID	INTEGER	The identifier of the user
INDEX_ID	INTEGER	The identifier of the index
COLUMN_ID	INTEGER	The column identifier
INDEX_COL_ORDER	INTEGER	The position of the column in the index
SORT_ORDER	CHAR(1)	The sort order
TABLE_ID	INTEGER	The table identifier

Column Information

USER_ID

This is the identifier of the owner of the index, and corresponds to a USER_ID in the SYS_USERS_ meta table.

INDEX_ID

This is the identifier of the index, and corresponds to an INDEX_ID in the SYS_INDICES_ meta table.

COLUMN_ID

This is the identifier of the column for which the index was created, and corresponds to a COLUMN_ID in the SYS_COLUMNS_ meta table.

INDEX_COL_ORDER

In the case of a composite index, because a single index spans multiple columns, this value indicates the position of the column in the index

SORT_ORDER

This indicates whether the index is arranged in ascending or descending order.

- A: Ascending order
- D: Descending order

TABLE_ID

This is the identifier of the table in which the index was created, and corresponds to a TABLE_ID value in the SYS_TABLES_ meta table.

Reference Tables

```
SYS_USERS_
SYS_TABLES_
SYS_COLUMNS_
SYS_INDICES_
```

SYS_INDEX_PARTITIONS_

This is the meta table for managing index partitions.

Column name	Type	Description
USER_ID	INTEGER	The user identifier
TABLE_ID	INTEGER	The tablespace identifier
INDEX_ID	INTEGER	The index identifier
TABLE_PARTITION_ID	INTEGER	The table partition identifier
INDEX_PARTITION_ID	INTEGER	The index partition identifier

Column name	Type	Description
INDEX_PARTITION_NAME	VARCHAR(128)	The index partition name
PARTITION_MIN_VALUE	VARCHAR(4000)	Reserved for future use
PARTITION_MAX_VALUE	VARCHAR(4000)	Reserved for future use
TBS_ID	INTEGER	The tablespace identifier
CREATED	DATE	The date and time at which the index parition is created
LAST_DDL_TIME	DATE	The time at which the index partition was most recently changed using a DDL statement

Column Information

USER_ID

This is the user identifier of the owner of the index. It corresponds to a USER_ID in the SYS_USERS_ meta table.

TABLE_ID

This is the identifier of the table in which the index is created. It is the same as a TABLE_ID value in the SYS_TABLES_ meta table.

INDEX_ID

This is the index identifier, and corresponds to an INDEX_ID in the SYS_INDICES_ meta table.

TABLE_PARTITION_ID

This is the table partition identifier.

INDEX_PARTITION_ID

This is the index partition identifier.

INDEX_PARTITION_NAME

This is the name of the index partition. It is specified by the user.

TBS_ID

This is the identifier of the tablespace in which the index is stored.

Reference Tables

```
SYS_USERS_
SYS_TABLES_
SYS_INDICES_
SYS_TABLE_PARTITIONS_
```

SYS_INDEX_RELATED_

This meta table contains information about the stored functions on which the function-based indexes are based.

Column name	Type	Description
USER_ID	INTEGER	The user identifier
TABLE_ID	INTEGER	The table identifier
INDEX_ID	INTEGER	The index identifier
RELATED_USER_ID	INTEGER	The identifier of the owner of the stored function referenced by the index
RELATED_PROC_NAME	VARCHAR(128)	) The name of the stored function referenced by the index

Column Infomation

USER_ID

This is the identifier of the owner of the index, and is identical to one USER_ID value in the SYS_USERS_ meta table.

TABLE_ID

This is the identifier of the table which defined the index, and is identical to one TABLE_ID value in the SYS_TABLES_ meta table.

INDEX_ID

This is the identifier of the index, and is identical to one INDEX_ID value in the SYS_INDICES_ meta table.

RELATED_USER_ID

This is the identifier of the owner of the stored function referenced by the index, and is identical to one USER_ID value in the SYS_USERS_ meta table.

RELATED_PROC_NAME

This is the name of the stored function referenced by the index, and is identical to one PROC_NAME value in the SYS_PROCEDURES_ meta table.

Reference Tables

```
SYS_USERS_
SYS_TABLES_
SYS_INDICES_
SYS_PROCEDURES_
```

SYS_INDICES_

This is the meta table that contains information about all indexes defined for all tables.

Column name	Type	Description
USER_ID	INTEGER	The user identifier
TABLE_ID	INTEGER	The table identifier
INDEX_ID	INTEGER	The index identifier
INDEX_NAME	VARCHAR(128)	The index name
INDEX_TYPE	INTEGER	The index type
IS_UNIQUE	CHAR(1)	Indicates whether the use of duplicate key values is allowed
COLUMN_CNT	INTEGER	The number of columns in the index
IS_RANGE	CHAR(1)	Indicates whether range scanning is possible using the index
IS_PERS	CHAR(1)	Indicates whether or not to permanently store the index
IS_DIRECTKEY	CHAR(1)	Indicates whether the index is a direct key index
TBS_ID	INTEGER	The tablespace identifier
IS_PARTITIONED	CHAR(1)	Indicates whether the index is partitioned
INDEX_TABLE_ID	INTEGER	Indicates the identifier for tables created by non-partitioned index of the partitioned table
CREATED	DATE	Indicates when the index was created
LAST_DDL_TIME	DATE	The time at which the index was most recently changed using a DDL statement

Column Information

USER_ID

This is the identifier of the owner of the index, and corresponds to a USER_ID value in the SYS_USERS_ meta table.

TABLE_ID

This is the identifier of the table in which the index was created, and corresponds to a TABLE_ID of the SYS_TABLES_ meta table.

INDEX_ID

This is an index identifier. It is automatically assigned by the system sequence.

INDEX_NAME

This is the name of the index.

INDEX_TYPE

This indicates the index type. A value of 1 indicates a B-TREE index, while a value of 2 indicates an R-TREE index.

IS_UNIQUE

This is indicates whether range scanning is possible using the index.

- T: Range scanning is possible.
- F: Range scanning is not possible.

COLUMN_CNT

This is the number of columns with which the index is associated.

IS_RANGE

This is indicates whether range scanning is possible using the index.

- T: Range scanning is possible.
- F: Range scanning is not possible.

IS_DIRECTKEY

This indicates whether the index is a direct key index.

- T: Direct key index
- F: Normal index

TBS_ID

This is the identifier of the tablespace in which the index was created.

IS_PARTITIONED

This indicates whether the index is partitioned. If it is 'T', the index is partitioned. If it is 'F', the index is not partitioned.

Reference Tables

```
SYS_USERS_
SYS_TABLES_
```

SYS_JOBS_

This meta table stores information about job objects.

Column name	Type	Description
JOB_ID	INTEGER	The job identifier
JOB_NAME	VARCHAR(128)	The job name
EXEC_QUERY	VARCHAR(1000)	The procedure registered for the job
START_TIME	DATE	The first time the job starts
END_TIME	DATE	The time the job ends
INTERVAL	INTEGER	The interval after which the job is to run
INTERVAL_TYPE	CHAR(2)	The unit of the interval after which the job is to run(YY, MM, DD, HH, MI)
STATE	INTEGER	The status of the job currently executing. 0: Is not running 1: Is running
LAST_EXEC_TIME	DATE	The last time the job was run
EXEC_COUNT	INTEGER	Execution frequency of the job
ERROR_CODE	CHAR(7)	An error code(NULL indicates success.)
IS_ENABLE	CHAR(1)	The status of job execution in the job scheduler. T: Possible to execute F: Impossible to execute
COMMENT	VARCHAR(4000)	An additional description for the job

Column Information

EXEC_QUERY

This indicates the procedure which is registered for the JOB and is executed.

INTERVAL_TYPE

This indicates the unit of time when an interval is set for the JOB. If a value exists in the INTERVAL column, this is the unit of the value.

- YY: Year
- MM: Mouth
- DD: Day
- HH: Hour
- MI: Minute

STATE

This indicates if a JOB is currently being executed or not.

- 0: Being executed
- 1: Not being executed

EXEC_COUNT

This indicates how many times a registered procedure has been executed since a JOB was created.

ERROR_CODE

This indicates the error code displaed if a procedure failed when the las JOB was executed. If succeeds, it is NULL.

IS_ENABLE

This indicates the possiblity of job execution in the job scheduler.

- T: Executable
- F: Not executable

COMMENT

This statement is used to describe a JOB. If the description is not delineated, NULL values are queried.

SYS_LIBRARIES_

This is the meta table that contains information about external library objects.

Column name	Type	Description
LIBRARY_ID	BIGINT	The library identifier
USER_ID	INTEGER	The user identifier
LIBRARY_NAME	VARCHAR(128)	The library name
FILE_SPEC	VARCHAR(4000)	The file path of the dynamic library
DYNAMIC	VARCHAR(1)	Reserved for future use
STATUS	VARCHAR(7)	Reserved for future use
CREATED	DATE	The time at which the library object was created
LAST_DDL_TIME	DATE	The time at which the library object was changed using a DDL statement for the last time.

Column Information

LIBRARY_ID

This is the library identifier and has a unique value within the system.

USER_ID

This is the user identifier of the library owner.

LIBRARY_NAME

This is the name of the library object and it has a unique value within the system.

FILE_SPEC

This is the file path of the dynamic library which the library object points to and it is a relative path for the default file path of the library($ALTIBASE_HOME/lib).

SYS_LOBS_

This is the meta table containing information about LOB columns defined in tables.

Column name	Type	Description
USER_ID	INTEGER	The user identifier
TABLE_ID	INTEGER	The table identifier
COLUMN_ID	INTEGER	The column identifier
TBS_ID	INTEGER	The tablespace identifier
LOGGING	CHAR(1)	This field is reserved for future use.
BUFFER	CHAR(1)	This field is reserved for future use.
IS_DEFAULT_TBS	CHAR(1)	Indicates whether a tablespace is designated for LOB column storage T: Specify F: Not to specify

Column Information

USER_ID

This is the identifier of the owner of the table to which the LOB column belongs, and corresponds to a USER_ID value in the SYS_USERS_ meta table.

TABLE_ID

This is the identifier of the table to which the LOB column belongs, and corresponds to a TABLE_ID value in the SYS_TABLES_ meta table.

COLUMN_ID

This is the LOB column identifier.

TBS_ID

This is the identifier of the tablespace to which the LOB column belongs.

IS_DEFAULT_TBS

This indicates whether a tablespace for storing a LOB column was specified by the user when the LOB column was created.

- T: Specify
- F Not to specify

For more detailed information, please refer to CREATE TABLE > LOB_STORAGE_CLAUSE statement in the *SQL reference*.

Reference Tables

```
SYS_USERS_
SYS_TABLES_
SYS_COLUMNS_
```

SYS_MATERIALIZED_VIEWS_

This is a meta table that contains information about materialized views.

Column name	Type	Description
USER_ID	INTEGER	The user identifier
MVIEW_ID	INTEGER	The materialized view identifier
MVIEW_NAME	VARCHAR(128)	The materialized view name
TABLE_ID	INTEGER	The table identifier
VIEW_ID	INTEGER	The view identifier
REFRESH_TYPE	CHAR(1)	The refresh type
REFRESH_TIME	CHAR(1)	The refresh time
CREATED	DATE	The time at which the table was created
LAST_DDL_TIME	DATE	The time when DDL was most recently used to make changes to a stored procedure
LAST_REFRESH_TIME	DATE	The last time the materialized view was refreshed

Column Information

USER_ID

This is the user identifier for the owner of the materialized view, it is identical to a USER_ID value of the SYS_USERS_ meta table.

MVIEW_ID

This is the materialized view identifier, automatically assigned by the database.

MVIEW_NAME

This is the name of the materialized view specified by the user.

TABLE_ID

This is the identifier for the table automatically created for the maintenance of the data of the materialized view. A table with the identical name of the materialized view can be checked by querying SYS_TABLES_ meta table with this identifier.

VIEW_ID

This is the identifier for the view automatically created for the maintenance of the data of the materialized view. This view can be looked up with this identifier at SYS_VIEWS_ meta table.

REFRESH_TYPE

This is the value taht indicates the refresh time of the materialized view.

- C: COMPLETE
- F: FAST
- R: FORCE

REFRESH_TIME

This is the value that indicates the refresh time of the materialized view.

- D: ON DEMAND
- C: ON COMMIT

CREATED

This is the date and time that the materialized view is created.

LAST_DDL_TIME

This is the date and time that the last alterations to the materialized view were made.

LAST_REFERESH_TIME

This is the date and time that the materialized view was refreshed lastly.

Reference Tables

```
SYS_USERS_
SYS_TABLES_
SYS_VIEWS_
SYS_VIEW_PARSE_
```

SYS_PACKAGES_

This meta table shows information about packages.

Column name	Type	Description
USER_ID	INTEGER	The user identifier of the package owner
PACKAGE_OID	BIGINT	The package identifier
PACKAGE_NAME	VARCHAR(128)	The package name
PACKAGE_TYPE	INTEGER	The package type. Indicates whether it is the package specification or the package body. 6: Package specification 7: Package body
AUTHID	INTEGER	The authority to execute the package 0: The definer 1: The current user
STATUS	INTEGER	The package status. If INVALID, the package is non-executable. 0: VALID 1: INVALID
CREATED	DATE	The time at which the package was created
LAST_DDL_TIME	DATE	The last time at which a DDL task was used to change the package

Column Information

USER_ID

This is the user identifier of the package owner; this value corresponds to one of the USER_ID values in the SYS_USERS_ meta table.

PACKAGE_OID

This is the package identifier; this value is automatically assigned by the system.

PACKAGE_NAME

This is the package name.

PACKAGE_TYPE

This is the value which indicates whether it is the package specification or the package body.

- 6: Package specification
- 7: Package body

AUTHID

This is the value which indicates authority to execute the package.

- 0: The definer
- 1: The current user

STATUS

This is the value which indicates whether or not the package is executable. 0(VALID) indicates that the package is executable.

- 0: VALID
- 1: INVALID

CREATED

This is the value which indicates the time at which the package was created.

LAST_DDL_TIME

This is the value which indicates the last time at which a DDL task was used to change the package.

Reference Table

```
SYS_USERS_
```

SYS_PACKAGE_PARAS_

This meta table contains information about subprogram(stored procedures and stored functions) parameters contained in packages.

Column name	Type	Description
USER_ID	INTEGER	The user identifier of the package owner
OBJECT_NAME	VARCHAR(128)	The subprogram name
PACKAGE_NAME	VARCHAR(128)	The package name
PACKAGE_OID	BIGINT	The package identifier
SUB_ID	INTEGER	The subprogram identifier
SUB_TYPE	INTEGER	The subprogram type 0: Procedure 1: Function
PARA_NAME	VARCHAR(128)	The name of the subprogram parameter

Column name	Type	Description
PARA_ORDER	INTEGER	The position of the parameter. The first parameter is assigned 1.
INOUT_TYPE	INTEGER	Whether the parameter is an Input, Output, or Input/Output parameter
DATA_TYPE	INTEGER	The parameter data type
LANG_ID	INTEGER	The language identifier of the parameter type
SIZE	INTEGER	The size of the parameter type
PRECISION	INTEGER	The precision of the parameter type
SCALE	INTEGER	The scale of the parameter type
DEFAULT_VAL	VARCHAR(4000)	The default value of the parameter

Column Information

USER_ID

This is the user identifier of the stored procedure or stored function owner; this value corresponds to one of the USER_ID values in the SYS_USERS_ meta table.

OBJECT_NAME

This is the subprogram name.

PACKAGE_NAME

This is the package name.

PACKAGE_OID

This is the package identifier and is identical to one of the values of PACKAGE_OID, which is the package specification in the SYS_PACKAGES_ meta table. This is the package identifier; this value corresponds to one of the PACKAGE_OID values in the package specification of the SYS_ PACKAGES _ meta table.

SUB_ID

This is the subprogram identifier. Inside a package, subprogram identifiers start from 1 and are assigned numbers in the order in which they are written.

SUB_TYPE

This is the value which indicates whether the subprogram is a stored procedure or a stored function

- 0: Procedure
- 1: Function

PARA_NAME

This is the name of the subprogram parameter.

PARA_ORDER

This is the value which indicates the nth order in which the given parameter was defined among other parameters.

INOUT_TYPE

This is the value which indicates whether the parameter of the stored procedure or stored function is an INPUT, OUTPUT or INPUT/OUTPUT parameter.

- 0: IN
- 1: OUT
- 2: IN OUT

DATA_TYPE

This is the identifier of the parameter data type. Please refer to the description of the DATA_TYPE column in the SYS_COLUMNS_ meta table for the value of the data type identifier.

For more detailed information about data types, please refer to Chapter 1.

LANG_ID

This is the column which displays information about the language properties of the character data types (CHAR, VARCHAR).

SIZE

This is the physical size of the data type.

PRECISION

This is the precision of the parameter data type, and is either specified by the user or assigned a default value by the system. For character data types, this value is the user-specified length of the character data type.

SCALE

This is the scale of the parameter data type, and is either specified by the user or assigned a default value by the system. Depending on the data type, the use of this value can be unnecessary.

For more detailed information about the scale and precision of data types, please refer to Chapter 1.

DEFAULT_VAL

This is the default parameter value specified by the user at the definition of a parameter.

Reference Tables

```
SYS_USERS_
SYS_PACKAGES_
```

SYS_PACKAGE_PARSE_

This meta table contains the statement text of user-defined packages.

Column name	Type	Description
USER_ID	INTEGER	The user identifier of the package owner
PACKAGE_OID	BIGINT	The package identifier
PACKAGE_TYPE	INTEGER	The package type. Indicates whether it is the package specification or the package body. 6: Package specification 7: Package body
SEQ_NO	INTEGER	The position of the record among multiple records of split and saved statements
PARSE	VARCHAR(100)	The statement which was split and saved

Column Information

USER_ID

This is the user identifier of the package owner; this value corresponds to one of the USER_ID values in the SYS_USERS_ meta table.

PACKAGE_OID

This is the package identifier; this value corresponds to one of the PACKAGE_OID values in the SYS_PACKAGES_ meta table.

PACKAGE_TYPE

This is the value which indicates whether it is the package specification or the package body.

- 6: Package specification
- 7: Package body

SEQ_NO

This is the value which indicates the nth order of each record among multiple records of split and saved package statements in SYS_PACKAGE_PARSE_.

PARSE

This is the string piece of the package statement. A CREATE PACKAGE statement can be made by searching for records with one PACKAGE_OID value and adding the PARSE values in the SEQ_NO order.

Reference Tables

```
SYS_USERS_
SYS_PACKAGES_
```

SYS_PACKAGE_RELATED_

This meta table contains information about tables, sequences, stored procedures, stored functions or views that are accessed by stored procedures and stored functions inside the package.

Column name	Type	Description
USER_ID	INTEGER	The user identifier of the package owner
PACKAGE_OID	BIGINT	The package identifier
RELATED_USER_ID	INTEGER	The owner identifier of the object referenced inside the package
RELATED_OBJECT_NAME	VARCHAR(128)	The object name referenced inside the package
RELATED_OBJECT_TYPE	INTEGER	The object type referenced inside the package

Column Information

USER_ID

This is the user identifier of the package owner; this value corresponds to one of the USER_ID values in the SYS_USERS_ meta table.

PACKAGE_OID

This is the package identifier; this value corresponds to the one of the PACKAGE_OID values in the SYS_ PACKAGES_ meta table.

RELATED_USER_ID

This is the user identifier of the owner of the object accessed by the stored procedure; this value corresponds to one of the USER_ID values in the SYS_USERS_ meta table.

RELATED_OBJECT_TYPE

This is the value which indicates the type of the object accessed by the stored procedure.

- 0: Stored procedure
- 1: Stored function

- 2: Table, sequence, view
- 3: Type set
- 4: Database link

Reference Tables

```
SYS_USERS_
SYS_PACKAGES_
SYS_TABLES_
```

SYS_PART_INDICES_

This is the meta table for managing partitioned indexes. It contains information about partitioned indexes for which IS_PARTITIONED in SYS_INDICES_ is set to 'T'.

Column name	Type	Description
USER_ID	INTEGER	The user identifier
TABLE_ID	INTEGER	The table identifier
INDEX_ID	INTEGER	The index identifier
PARTITION_TYPE	INTEGER	The partition type
IS_LOCAL_UNIQUE	CHAR(1)	Indicates whether an index is a local unique index

Column Information

USER_ID

This is the user identifier of the owner of the index, and corresponds to a USER_ID in the SYS_USERS_ meta table.

TABLE_ID

This is the identifier of the table for which the index was created, and corresponds to a TABLE_ID value in the SYS_TABLES_ meta table.

INDEX_ID

This is the index identifier. It corresponds to an INDEX_ID value in the SYS_INDICES_ meta table.

PARTITION_TYPE

This indicates whether the partition type is LOCAL or GLOBAL. However, because the GLOBAL partition type is not supported at present, it is always 0.

- 0: LOCAL
- 1: GLOBAL

IS_LOCAL_UNIQUE

This indicates whether an index is a local unique index, and can be 'T' or 'F'.

- T: A local unique index
- F: Not a local unique index

Reference Tables

```
SYS_USERS_
SYS_TABLES_
SYS_INDICES_
```

SYS_PART_KEY_COLUMNS_

This meta table shows information about the partitioning key columns for the partitioned objects

Column name	Type	Description
USER_ID	INTEGER	The user identifier
PARTITION_OBJ_ID	INTEGER	The partitioned object identifier
COLUMN_ID	INTEGER	The column identifier
OBJECT_TYPE	INTEGER	The object type
PART_COL_ORDER	INTEGER	The position of the column in the partitioning key (starting with 0)

Column Information

USER_ID

This is the identifier of the owner of the partitioned table or index. It corresponds to a USER_ID value in the SYS_USERS_ meta table.

PARTITION_OBJ_ID

This is the identifier of a partitioned object, and corresponds to a TABLE_ID value in the SYS_PART_TABLES_ meta table or INDEX_ID value in the SYS_PART_INDICES_ meta table.

COLUMN_ID

This is the identifier of the column in the partitioning key, and corresponds to a COLUMN_ID value in the SYS_COLUMNS_ meta table.

OBJECT_TYPE

This identifies the type of the object.

- 0: TABLE
- 1: INDEX

PART_COL_ORDER

This is the position of the column in the partitioning key (starting with 0).

Reference Tables

```
SYS_PART_INDICES_
SYS_TABLE_PARTITIONS_
SYS_COLUMNS_
```

SYS_PART_LOBS_

This is a meta table for managing LOB columns for respective partitions.

Column name	Type	Description
USER_ID	INTEGER	The user identifier
TABLE_ID	INTEGER	The table identifier
PARTITION_ID	INTEGER	The partition identifier
COLUMN_ID	INTEGER	The column identifier
TBS_ID	INTEGER	The tablespace identifier
LOGGING	CHAR(1)	This field is reserved for future use.
BUFFER	CHAR(1)	This field is reserved for future use.

Column Information

USER_ID

This is the identifier of the owner of the table to which the LOB column belongs, and corresponds to a USER_ID value in the SYS_USERS_ meta table.

TABLE_ID

This is the identifier of the table to which the LOB column belongs, and corresponds to a TABLE_ID value in the SYS_TABLES_ meta table.

PARTITION_ID

This is the identifier of the partition in which the LOB column is stored.

COLUMN_ID

This is the LOB column identifier.

TBS_ID

This is the identifier of the tablespace to which the LOB column belongs.

Reference Tables

```
SYS_USERS_
SYS_TABLES_
SYS_PART_TABLES_
SYS_COLUMNS_
```

SYS_PART_TABLES_

This is the meta table for the management of partitioned tables. The table information in SYS_PART_TABLES_ is information about partitioned tables for which IS_PARTITIONED in SYS_TABLES_ is set to 'T'.

Column name	Type	Description
USER_ID	INTEGER	The user identifier
TABLE_ID	INTEGER	The table identifier
PARTITION_METHOD	INTEGER	The partitioning method
PARTITION_KEY_COUNT	INTEGER	The number of partition key columns
ROW_MOVEMENT	CHAR(1)	Indicates whether updated records can be moved between partitions

Column Information

USER_ID

This is the identifier of the owner of the index, and corresponds to a USER_ID value in the SYS_USERS_ meta table.

TABLE_ID

This is the identifier of the table in which the index was created, and corresponds to a TABLE_ID value in the SYS_TABLES_ meta table.

PARTITION_METHOD

This indicates the partitioning method.

- 0: RANGE
- 1: HASH
- 2: LIST
- 3: RANGE PARTITIONING USING HASH

ROW_MOVEMENT

This indicates whether it is permissible for records that have been updated to be moved to other partitions when the value of a partition key column is updated.

- T: Movement of updated records between partition is permitted
- F: Movement of updated records between partition is forbidden

Reference Tables

```
SYS_USERS_
SYS_TABLES_
```

SYS_PASSWORD_HISTORY_

This meta table stores alterations made to user passwords that have been assigned a password policy.

Column name	Type	Description
USER_ID	INTEGER	The user identifier
PASSWORD	VARCHAR(256)	The user password
PASSWORD_DATE	DATE	The date one which alterations are made to the user password

SYS_PASSWORD_LIMITS_

This meta table stores specified password management policies at user creation and account status quo.

Column name	Type	Description
USER_ID	INTEGER	The user identifier
USER_NAME	VARCHAR(128)	The user name
ACCOUNT_STATUS	VARCHAR(30)	Indicates account status quo: EXPIRED EXPIRED(GRACE) LOCKED(TIMED) LOCKED EXPIRED & LOCKED(TIMED) EXPIRED(GRACE) & LOCKED(TIMED) EXPIRED & LOCKED EXPIRED(GRACE) & LOCKED
REMAIN_GRACE_DAY	VARCHAR(10)	The grace period remaining after password expiry date
FAILED_LOGIN_ATTEMPTS	VARCHAR(10)	The maximum number of times login failure is permitted

Column name	Type	Description
PASSWORD_LOCK_TIME	VARCHAR(10)	The amount of time needed to elapse for a locked account to become unlocked
PASSWORD_LIFE_TIME	VARCHAR(10)	The password validity period
PASSWORD_GRACE_TIME	VARCHAR(10)	The grace period following password expiry date
PASSWORD_REUSE_TIME	VARCHAR(10)	The amount of time needed to elapse for identical passwords to be available for reuse
PASSWORD_REUSE_MAX	VARCHAR(10)	The number of times identical passwords are available for reuse
PASSWORD_VERIFY_FUNCTION	VARCHAR(128)	The CALLBACK function for verifying passwords

SYS_PRIVILEGES_

This meta table contains information about the kinds of privileges supported by Altibase. For more detailed information, please refer to the descriptions of database privileges and of the GRANT statement in the Reference

Column name	Type	Description
PRIV_ID	INTEGER	The privilege identifier
PRIV_TYPE	INTEGER	The privilege type
PRIV_NAME	VARCHAR(128)	The privilege name

Column Information

PRIV_ID

This is the privilege identifier. It is defined internally by the system.

PRIV_TYPE

This indicates the type of privilege.

- 1: Indicates an object privilege
- 2: Indicates a system privilege

PRIV_NAME

This is the name of the privilege.

SYS_PROCEDURES_

This table is for storing information about stored procedures and stored functions, such as the stored procedure name, return type, number of parameters, whether it can be executed, etc.

Column name	Type	Description
USER_ID	INTEGER	The identifier of the owner of the stored procedure
PROC_OID	BIGINT	The identifier of the stored procedure
PROC_NAME	VARCHAR(128)	The name of the stored procedure
OBJECT_TYPE	INTEGER	Indicates whether the object is a stored procedure, stored function, or type set
STATUS	INTEGER	Indicates the status of the object. The object cannot be executed if it is INVALID. 0: VALID 1: INVALID
AUTHID	INTEGER	The authority to execute the package - 0: The definer - 1: The current user
PARA_NUM	INTEGER	The number of parameters for the stored procedure
RETURN_DATA_TYPE	INTEGER	The return data type for the stored function
RETURN_LANG_ID	INTEGER	The return type language identifier
RETURN_SIZE	INTEGER	The size of the stored function return data type
RETURN_PRECISION	INTEGER	The precision of the stored function return data type
RETURN_SCALE	INTEGER	The size of the stored function return data type
PARSE_NO	INTEGER	The number of records containing statement fragments stored in SYS_PROC_PARSE_ for the procedure
PARSE_LEN	INTEGER	The total length of the procedure statement stored in SYS_PROC_PARSE_
CREATED	DATE	The date on which the object was created
LAST_DDL_TIME	DATE	The time when DDL was most recently used to make changes to a stored procedure

Column Information

USER_ID

This is the identifier of the owner of the stored procedure or stored function, and corresponds to a USER_ID value in the SYS_USERS_ meta table.

PROC_OID

This is the identifier of the stored procedure or stored function, and is automatically assigned by the system.

PROC_NAME

This is the name of the stored procedure or stored function.

OBJECT_TYPE

This value allows stored procedures to be distinguished from stored functions. Stored functions differ from stored procedures in that they return a value.

- 0: Stored procedure
- 1: Stored function
- 3: Type set

STATUS

This value indicates whether a stored procedure or function may be executed. A value of 0 (VALID) indicates that it can be executed.

If a DDL statement is executed on an object that is accessed by a stored procedure or stored function, the stored procedure or stored function will become invalid. For example, if a new column is added to a table that is accessed by a stored procedure, the stored procedure will need to be re-compiled before it can be deemed VALID and executed. The status values are as follows:

- 0: VALID
- 1: INVALID

AUTHID

This is the value which indicates the authority to execute the procedure or function.

- 0: DEFINER
- 1: CURRENT_USER

PARA_NUM

This indicates the number of parameters defined for a stored procedure or stored function.

RETURN_DATA_TYPE

This is the data type identifier for the return value of a stored function. Information about data type identifiers can be found in the DATA_TYPE column of the SYS_COLUMNS_ meta table.

For more information about data types, please refer to Chapter1: Data Types.

RETURN_LANG_ID

This column contains information about the language properties of the character data types (CHAR, VARCHAR).

RETURN_SIZE

This is the physical size of the return data type.

RETURN_PRECISION

This is the precision of the return data type, which is either defined by the user or set based on the system default. For character types, it is the length of the user-defined character type.

RETURN_SCALE

This is the scale of the return data type, which is either defined by the user or set as the system default. Depending on the type, this value may not be used.

For more information about data type precision and scale, please refer to Chapter1: Data Types.

PARSE_NO

Stored procedure and stored function statements are divided into multiple records containing text fragments and stored in the SYS_PROC_PARSE_ meta table. This value indicates the number of records used to store a stored procedure or function.

PARSE_LEN

Stored procedure and stored function statements are divided into multiple records containing text fragments and stored in the SYS_PROC_PARSE_ meta table. This value indicates the overall length of the statement.

LAST_DDL_TIME

This is the most recent time at which a DDL statement was used to make changes to a stored procedure.

Reference Table

```
SYS_USERS_
```

SYS_PROC_PARAS_

This meta table contains information about the parameters of stored procedures and stored functions.

Column name	Type	Description
USER_ID	INTEGER	The identifier of the owner of the stored procedure
PROC_OID	BIGINT	The identifier of the stored procedure
PARA_NAME	VARCHAR(128)	The parameter name
PARA_ORDER	INTEGER	The parameter order. The first parameter is assigned the number 1.
INOUT_TYPE	INTEGER	Whether the parameter is an Input, Output, or Input/Output parameter
DATA_TYPE	INTEGER	The data type of the parameter
LANG_ID	INTEGER	The language identifier for the parameter type
SIZE	INTEGER	The size of the parameter type
PRECISION	INTEGER	The precision of the parameter type
SCALE	INTEGER	The scale of the parameter type
DEFAULT_VAL	VARCHAR(4000)	The default value for the parameter

Column Information

USER_ID

This is the identifier of the user who is the owner of the stored procedure or the stored function, and corresponds to a USER_ID in the SYS_USERS_ meta table.

PROC_OID

This is the identifier of the stored procedure or stored function, and corresponds to a PROC_ID in the SYS_PROCEDURES_ meta table

PARA_NAME

This is the parameter name.

PARA_ORDER

When there are multiple parameters, this value indicates the position of the parameter in the defined parameter order.

INOUT_TYPE

This value indicates whether the parameter for the stored procedure or stored function is an input, output, or input/output parameter.

- 0: IN
- 1: OUT
- 2: IN OUT

DATA_TYPE

This is the data type identifier for the parameter. The DATA_TYPE column in the SYS_COLUMNS_ meta table contains information about data type identifiers.

For more information about data types, please refer to Chapter1: Data Types.

LANG_ID

This column displays the language properties for character type parameters (CHAR and VARCHAR).

SIZE

This is the physical size of the data type.

PRECISION

This is the precision of the parameter, which is either determined by the user or set based on the system default. The precision (length) of character data types is defined by the user.

SCALE

This is the scale of the parameter, which is either determined by the user or set to the system default. Depending on the data type, this value may not be used.

For more information about the scale and precision of data types, please refer to Chapter1: Data Types.

DEFAULT_VAL

When a parameter is defined, this is the user-defined default parameter value.

Reference Tables

```
SYS_USERS_
SYS_PROCEDURES_
```

SYS_PROC_PARSE_

This meta table contains the text constituting user-defined stored procedures and stored functions.

Column name	Type	Description

Column name	Type	Description
USER_ID	INTEGER	The identifier of the owner of the stored procedure or stored function
PROC_OID	BIGINT	The object identifier of the stored procedure
SEQ_NO	INTEGER	The position of the record among multiple records for a statement that was split and then saved
PARSE	VARCHAR(100)	A fragment of the text of the stored procedure or stored function

Column Information

USER_ID

This is the identifier of the owner of the stored procedure or stored function, and corresponds to a USER_ID in the SYS_USERS_ meta table.

PROC_OID

This is the identifier of the stored procedure or the stored function, and corresponds to a PROC_ID in the SYS_PROCEDURES_ meta table.

SEQ_NO

When the information for a statement for one stored procedure is saved across multiple records in SYS_PROC_PARSE_, this is the sequential position of an individual record.

PARSE

This is a line of text belonging to the stored procedure or stored function. An entire statement of a stored procedure can be re-created by retrieving all records that correspond to a single PROC_OID value and combining the PARSE values in order according to the SEQ_NO values.

Reference Tables

```
SYS_USERS_
SYS_PROCEDURES_
```

SYS_PROC_RELATED_

This table contains information about tables, sequences, stored procedures, stored functions, and views accessed by a stored procedure or stored function

Column name	Type	Description
USER_ID	INTEGER	The identifier of the owner of the stored procedure

Column name	Type	Description
PROC_OID	BIGINT	The identifier of the stored procedure
RELATED_USER_ID	INTEGER	The identifier of the owner of an object referenced within a stored procedure
RELATED_OBJECT_NAME	VARCHAR(128)	The name of an object referenced within a stored procedure
RELATED_OBJECT_TYPE	INTEGER	The type of an object referenced within a stored procedure

In the case where stored procedure PROC1 performs an INSERT on table t1, the identifiers for the owner of the stored procedure PROC1 and for the stored procedure itself would be stored in USER_ID and PROC_OID respectively, the identifiers for the owner of table t1 and for the table itself would be stored in RELATED_USER_ID and RELATED_OBJECT_NAME respectively, and the number 2 (signifying a table) would be stored in RELATED_OBJECT_TYPE.

Column Information

USER_ID

This is the identifier of the owner of the stored procedure or the stored function, and corresponds to a USER_ID in the SYS_USERS_ meta table.

PROC_OID

This is the identifier of the stored procedure or the stored function, and corresponds to a PROC_ID in the SYS_PROCEDURES_ meta table.

RELATED_USER_ID

This is the identifier of the owner of the object accessed by the stored procedure, and corresponds to a USER_ID in the SYS_USERS_ meta table.

RELATED_OBJECT_NAME

This is the name of the object accessed by the stored procedure.

RELATED_OBJECT_TYPE

This is the type of the object accessed by the stored procedure. The possible values are as follows:

- 0: Stored procedure
- 1: Stored function
- 2: Table, Sequence, View
- 3: Type set
- 4: Database link

Reference Tables

```
SYS_USERS_
SYS_PROCEDURES_
SYS_TABLES_
```

SYS_RECYCLEBIN_

This table contains information about tables in the recycle bin. If the value for the RECYCLEBIN_ENABLE property is set to 1, this table contains information about tables moved to the recycle bin using the DROP statement.

Column name	Type	Description
USER_NAME	VARCHAR(128)	The table owner
TABLE_NAME	VARCHAR(128)	The table name generated by the system so that the table can be managed in the recycle bin when it is dropped. A table with the same name can be dropped multiple times; different names are generated so that tables can be managed in the recycle bin.
ORIGINAL_TABLE_NAME	VARCHAR(128)	The table name before it was dropped.
TBS_NAME	VARCHAR(128)	The tablespace name in which the table is saved.
MEMORY_SIZE	BIGINT	The total memory space occupied by the memory table that was dropped.
DISK_SIZE	BIGINT	The total disk space occupied by the disk table that was dropped.
DROPPED	DATE	The date at which the table was dropped.

Column Information

TABLE_NAME

This is the table name generated by the system when it is moved to the recycle bin. If tables with the same name (ORIGINAL_TABLE_NAME) are dropped multiple times, new names are generated for the tables in the recycle bin.

SYS_REPLICATIONS_

This meta table contains information related to replication.

Column name	Type	Description
LAST_USED_HOST_NO	INTEGER	The most recently used remote server

Column name	Type	Description
HOST_COUNT	INTEGER	The number of remote servers
IS_STARTED	INTEGER	Whether replication is active
XSN	BIGINT	The Restart SN (Sequence Number), i.e. the SN from which the Sender will resume transmission of XLogs[13]
ITEM_COUNT	INTEGER	The number of replication target tables
CONFLICT_RESOLUTION	INTEGER	The replication conflict resolution method
REPL_MODE	INTEGER	The default replication mode
ROLE	INTEGER	The role of the sender thread
OPTIONS	INTEGER	A flag for additional replication features
INVALID_RECOVERY	INTEGER	Whether replication recovery is possible
REMOTE_FAULT_DETECT_TIME	DATE	The time at which a fault was detected on a remote server
GIVE_UP_TIME	DATE	The time at which replication was most recently abandoned
REPLICATION_NAME	VARCHAR(40)	The name of the replication object
GIVE_UP_XSN	BIGINT	The XSN at which replication was most recently abandoned
PARALLEL_APPLIER_COUNT	INTEGER	The number of appliers
REMOTE_XSN	BIGINT	The most recently processed SN on the remote server.
APPLIER_INIT_BUFFER_SIZE	BIGINT	Initial size of applier buffer

[13] SN: The identification number of the log record

Column Information

REPLICATION_NAME

This is the name of the replication object, and is set by the user when the replication object is created.

LAST_USED_HOST_NO

This is the number of the most recently used remote server, and corresponds to a HOST_NO in the SYS_REPL_HOSTS_ meta table.

HOST_COUNT

This is the number of remote servers involved in replication, and is equal to the number of IP addresses stored in SYS_REPL_HOSTS_.

IS_STARTED

Indicates whether replication is active.

- 0: suspended
- 1: replication active

XSN

This indicates the SN from which the Sender thread must begin sending logs when replication is started.

ITEM_COUNT

This is the number of replication target tables. This number corresponds to the number of records in the SYS_REPL_ITEMS_ meta table for this replication object, with one record corresponding to each of these tables.

CONFLICT_RESOLUTION

This describes the replication conflict resolution method.

- 0: Default
- 1: Act as the Master server
- 2: Act as the Slave server

Please refer to the *Replication Manual* for more detailed information about replication conflict resolution methods.

REPL_MODE

This is the default replication mode, which is set when the replication object is created.

- 0: LAZY MODE (Default)
- 2: EAGER MODE

The default replication mode is used if the ALTER SESSION SET REPLICATION statement is not used to set the replication mode for a session.

For more detailed information about the default replication mode, please refer to the *Replication Manual*, and for detailed information about the ALTER SESSION SET REPLICATION statement, please refer to the *SQL Reference*.

ROLE

This indicates the role of the Sender thread.

- 0: Replication
- 1: Log Analyzer

- 2: Propagable Logging (Replication propagable logs)
- 3: Propagation (Send propagable logs)

For more detailed information, please refer to the *Log Analyzer User's Manual*.

OPTIONS

This flag indicates whether to use the recovery and offline options, which are extra replication features. The replication option types are as in the following. Each option is controlled by binary number and expressed as a decimal number. If more than two options are used, the sum of the binary numbers of each option is returned as decimal numbers.

- 0(00000): Do not use the recovery or offline option
- 1(00001): Use the recovery option
- 2(00010): Use the offline option
- 4(00100): Use the gapless option
- 8(01000): Use the parallel applier option
- 16(10000): Use the replication transaction grouping option

INVALID_RECOVERY

This value indicates whether recovery using replication is possible.

- 0: replication-based recovery is possible.
- 1: replication-based recovery is not possible.

REMOTE_FAULT_DETECT_TIME

This is the time at which a fault was detected on a remote server while replication was running.

GIVE_UP_TIME

This is the time at which replication was most recently abandoned, i.e. the time at which the replication Sender most recently gave up on replication.

GIVE_UP_XSN

This is the XSN at which replication was most recently abandoned.

PARALLEL_APPLIER_COUNT

This is the number of parallel applier.

REMOTE_XSN

This is the most recently processed SN on the remote server. When the Sender is restarted, the log with the SN smaller than the corresponding REMOTE_XSN is not sent but skipped.

APPLIER_INIT_BUFFER_SIZE

This property specifies the initial buffer size of the parallel applier when replication is performed with the receiver applier option enabled. The number of queues in the parallel applier is set to the value divided by XLog Size.

```
( applier queue size = applier_init_buffer_size / xlog size )
```

If the number of parallel applier queues is less than the value of the property REPLICATION_RECEIVER_APPLIER_QUEUE_SIZE, then the number of parallel applier queues is set to the value specified in the property REPLICATION_RECEIVER_APPLIER_QUEUE_SIZE.

Example

<Example> The following is an example of returning values when using the replication gapless option and parallel applier option together on a created replication rep1.

```
iSQL> alter replication rep1 set gapless enable;
Alter success.
iSQL> alter replication rep1 set parallel 4;
Alter success.
iSQL> select options from system_.sys_replications_;
OPTIONS
--------------
12
1 row selected.
```

SYS_REPL_HOSTS_

This meta table contains information related to remote servers defined in replication objects.

Column name	Type	Description
HOST_NO	INTEGER	The host identifier
REPLICATION_NAME	VARCHAR(40)	The replication name
HOST_IP	VARCHAR(64)	The IP address of the remote server
PORT_NO	INTEGER	The replication port number on the remote server
CONN_TYPE	VARCHAR(20)	The remote server connection method
IB_LATENCY	VARCHAR(10)	The RDMA_LATENCY option value for rsocket.

Column Information

HOST_NO

This is the serial number of the remote server, which is automatically assigned by the system sequence.

REPLICATION_NAME

This is the name of the replication object set by the user, and corresponds to a REPLICATION_NAME in the SYS_REPLICATIONS_ meta table.

HOST_IP

This is the IP address of the remote server.

PORT_NO

This is the replication port number on the remote server.

CONN_TYPE

This shows the remote server connection method.

- TCP
- Unix Domain
- InfiniBand(IB)

IB_LATENCY

This is the value of the RDMA_LATENCY option of rsocket when using InfiniBand, This value is N/A when CONN_TYPE is not IB.

Reference Table

```
SYS_REPLICATIONS_
```

SYS_REPL_ITEMS_

This meta table contains information about replication target tables.

Column name	Type	Description
REPLICATION_NAME	VARCHAR(40)	The replication name
TABLE_OID	BIGINT	The table object identifier
LOCAL_USER_NAME	VARCHAR(128)	The name of a user owning a target table on the local server
LOCAL_TABLE_NAME	VARCHAR(128)	The name of a target table on the local server
LOCAL_PARTITION_NAME	VARCHAR(128)	The name of a partition on the local server

Column name	Type	Description
REMOTE_USER_NAME	VARCHAR(128)	The name of a user owning a target table on the remote server
REMOTE_TABLE_NAME	VARCHAR(128)	The name of a target table on the remote server
REMOTE_PARTITION_NAME	VARCHAR(128)	The name of a partition on the remote server
IS_PARTITION	CHAR(1)	Whether or not a table is partitioned
INVALID_MAX_SN	BIGINT	The highest log SN to skip
CONDITION	VARCHAR(1000)	Deprecated
REPLICATION_UNIT	CHAR(1)	The replication unit
IS_CONDITION_SYNCED	INTEGER	Whether or not a replication is conditional synced

One replication object can pertain to more than one table, and SYS_REPL_ITEMS_ has a record for each of these tables. For example, if a replication pertains to 10 tables, this meta table will contain 10 records pertaining to this replication.

Column Information

REPLICATION_NAME

This is the name of the replication object, which is defined by the user, and corresponds to a REPLICATION_NAME in the SYS_REPLICATIONS_ meta table

TABLE_OID

This is the identifier of the replication target table or partition, and corresponds to a TABLE_OID value in the SYS_TABLES_ meta table or a PARTITION_OID value in the SYS_TABLES_PARTITIONS meta table.

LOCAL_USER_NAME

This is the user name of the owner of the replication target table in the local system, and corresponds to a USER_NAME in the SYS_USERS_ meta table.

LOCAL_TABLE_NAME

This is the name of the replication target table in the local system, and corresponds to a TABLE_NAME in the SYS_TABLES_ meta table.

LOCAL_PARTITION_NAME

This is the name of the replication target partition on the local server.

REMOTE_USER_NAME

This is the user name of the owner of the replication target table in the remote system, and corresponds to a USER_NAME in the SYS_USERS_ meta table.

REMOTE_TABLE_NAME

This is the name of the replication target table in the remote system, and corresponds to a TABLE_NAME in the SYS_TABLES_ meta table.

REMOTE_PARTITION_NAME

This is the name of the replication target partition on the remote server.

IS_PARTITION

This is an identifier indicating whether a table is partitioned. If it is 'Y', the table is partitioned. If it is 'N', the table is not partitioned.

INVALID_MAX_SN

If DDL statements or Sync operations are executed on replication target tables, the most recently recorded SN is saved here. Table logs up to this SN are skipped when the table is replicated.

REPLICATION_UNIT

This is the unit of the replication target item. One of the following two values is indicated in this column.

- T: The replication target item is a table.
- P: The replication target item is a partition.

IS_CONDITION_SYNCED

Whether or not a replication is conditional synced |

Reference Tables

```
SYS_REPLICATIONS_
SYS_USERS_
SYS_TABLES_
```

SYS_REPL_OFFLINE_DIR_

This meta table stores log directory information related to the offline replication option.

Column name	Type	Description
REPLICATION_NAME	VARCHAR(40)	The replication name
LFG_ID	INTEGER	The identifier of the log file group
PATH	VARCHAR(512)	The offline log path

Column Information

REPLICATION_NAME

This is the user-defined replication name. It corresponds to a REPLICATION_NAME in the SYS_REPLICATIONS_ meta table.

LFG_ID

This is the identifier for the LFG which defalut value is '0'.

PATH

This is the absolute path in the system where the log file is saved.

SYS_REPL_OLD_CHECKS_

This meta table is for storing information about replication target columns that is being replicated by replication sender thread and has CHECK constraints.

Column name	Type	Description
REPLICATION_NAME	VARCHAR(40)	The name of the replication object
TABLE_OID	BIGINT	The table object identifier
CONSTRAINT_ID	INTEGER	The identifier of CHECK constraint
CHECK_NAME	VARCHAR(40)	The name of the CHECK constraint
CONDITION	VARCHAR(4000)	The character string condition of the CHECK constraint

Column Information

REPLICATION_NAME

This is the name of the replication object set by the user, and can be found in the SYS_REPLICATIONS_ meta table.

TABLE_OID

This is the identifier for a replication target table currently being used by the replication sender thread. Its value may not be found in SYS_TABLES_ meta table if this table does not exist when the replication sender thread is processing replication log.

CONSTRAINT_ID

This is the identifier of the CHECK constraint that is being processed by replication sender thread, and corresponds to a CONSTRAINT_ID value in the SYS_CONSTRAINTS_ meta table.

Its value cannot be found in SYS_CONSTRAINTS_ if this CHECK constraint was removed while the replication sender thread was processing the replication log.

CHECK_NAME

This is the name of the CHECK constraint that replication sender thread is currently using. It corresponds to a CONSTRAINT_NAME value in the SYS_CONSTRAINTS_ meta table.

Its value cannot be found in SYS_CONSTRAINTS_ if this CHECK constraint was removed while the replication sender thread was processing the replication log.

CONDITION

This is the character string condition of the CHECK constraint that replication sender thread is currently using. It corresponds to a CHECK_CONDITION value in the SYS_CONSTRAINTS_ meta table.

Its value cannot be found in SYS_CONSTRAINTS_ if this CHECK constraint was removed while the replication sender thread was processing the replication log.

Reference Tables

```
SYS_REPLICATIONS_
SYS_TABLES_
SYS_CONSTRAINTS_
```

SYS_REPL_OLD_CHECK_COLUMNS_

This meta table is for storing information about CHECK constraints on replication target column that replication sender thread is currently processing.

Column name	Type	Description
REPLICATION_NAME	VARCHAR(40)	The name of the replication object
TABLE_OID	BIGINT	The object identifier of the table
CONSTRAINT_ID	INTEGER	The identifier of CHECK constraint
COLUMN_ID	INTEGER	The identifier of column that has CHECK constraint

Column Information

REPLICATION_NAME

This is the name of the replication object set by the user, and can be found in the SYS_REPLICATIONS_ meta table.

TABLE_OID

This is the identifier for a replication target table currently being used by the replication sender thread. Its value may not be found in SYS_TABLES_ meta table if this table does not exist when the replication sender thread is processing replication log.

CONSTRAINT_ID

This is the identifier of the CHECK constraint that is being processed by replication sender thread, and corresponds to a CONSTRAINT_ID value in the SYS_CONSTRAINTS_ meta table.

Its value cannot be found in SYS_CONSTRAINTS_ if this CHECK constraint was removed while the replication sender thread was processing the replication log.

COLUMN_ID

This is the identifier of the column that is currently being processed by replication sender thread and has CHECK constraint. It corresponds to a COLUMN_ID value in the SYS_COLUMNS_ meta table. Its value cannot be found in SYS_COLUMNS_ if this CHECK constraint was removed while the replication sender thread was processing the replication log.

Reference Tables

```
SYS_REPLICATIONS_
SYS_TABLES_
SYS_CONSTRAINTS_
SYS_COLUMNS_
```

SYS_REPL_OLD_COLUMNS_

This meta table is for storing information about columns that are currently replicated by the replication Sender thread.

Column name	Type	Description
REPLICATION_NAME	VARCHAR(40)	The name of the replication object
TABLE_OID	BIGINT	The object identifier of the table
COLUMN_NAME	VARCHAR(128)	The column name
MT_DATATYPE_ID	INTEGER	The data type identifier
MT_LANGUAGE_ID	INTEGER	The language identifier
MT_FLAG	INTEGER	An internal flag
MT_PRECISION	INTEGER	The number of digits
MT_SCALE	INTEGER	The number of digits to the right of the decimal point
MT_ENCRYPT_PRECISION	INTEGER	The number of digits in an encrypted column
MT_POLICY_NAME	VARCHAR(16)	The name of the policy used for an encrypted column
SM_ID	INTEGER	The column identifier

Column name	Type	Description
SM_FLAG	INTEGER	An internal flag
SM_OFFSET	INTEGER	The internal offset
SM_VARORDER	INTEGER	Indicates the order of stored columns by variables method in a table. Exceptionally, VARORDER is not given to the geometry data type. (Default : 0)
SM_SIZE	INTEGER	The internal size
SM_DIC_TABLE_OID	BIGINT	For a compressed column, the OID of the dictionary table
SM_COL_SPACE	INTEGER	The tablespace identifier
QP_FLAG	INTEGER	The internal flag
DEFAULT_VAL	VARCHAR(4000)	The default value of the column

Column Information

REPLICATION_NAME

This is the replication name, which is specified by the user. It corresponds to a REPLICATION_NAME in the SYS_REPLICATIONS_ meta table.

TABLE_OID

This is the identifier for a replication target table currently being used by the replication Sender thread. Its value may not correspond to any TABLE_OID value in SYS_TABLES_.

COLUMN_NAME

This is the name of a column currently being replicated by the replication Sender thread.

MT_DATATYPE_ID

This is the data type identifier, and is an internal value.

MT_LANGUAGE_ID

This is the language identifier, and is an internal value.

MT_FLAG

This is an internal flag used by Altibase server.

MT_PRECISION

For a numeric type column, this is the number of digits in the column.

MT_SCALE

For a numeric type column, this is the number of digits to the right of the decimal point in the column.

MT_ENCRYPT_PRECISION

For an encrypted numeric type column, this is the number of digits in the column.

MT_POLICY_NAME

For an encrypted column, this is the name of the policy used for the column.

SM_ID

This is the column identifier. Column identifiers start with 0.

SM_FLAG

This is a flag internally used by Altibase.

SM_OFFSET

This is an offset value internally used by Altibase.

SM_VARORDER

This indicates the order of a column among the columns stored with the variable method within a table. Exceptionally, VARORDER is not given to the geometry data type (Default value:0).

SM_SIZE

This is a size value internally used by Altibase server.

SM_DIC_TABLE_OID

For a compressed column, this is the OID of the dictionary table in which data of the compressed column is actually saved.

SM_COL_SPACE

This is the identifier of the tablespace in which the column data is saved.

QP_FLAG

This is the flag used internally by the Altibase server.

DEFAULT_VAL

The default value of the column is saved as a string and is used internally by the Altibase server.

Reference Tables

```
SYS_REPL_OLD_ITEMS_
SYS_REPL_OLD_INDICES_
SYS_REPL_OLD_INDEX_COLUMNS_
```

SYS_REPL_OLD_INDEX_COLUMNS_

This meta table is for storing information about columns currently being replicated by the replication Sender thread.

Column name	Type	Description
REPLICATION_NAME	VARCHAR(40)	The replication name
TABLE_OID	BIGINT	The table object identifier
INDEX_ID	INTEGER	The index identifier
KEY_COLUMN_ID	INTEGER	The column identifier
KEY_COLUMN_FLAG	INTEGER	An internal flag
COMPOSITE_ORDER	INTEGER	The position of the column on which the index is based

Column Information

REPLICATION_NAME

This value corresponds to a REPLICATION_NAME in the SYS_REPLICATIONS_ meta table, and is the user-defined replication name.

TABLE_OID

This is the identifier of a table currently being replicated by the replication Sender thread. Its value may not correspond to any TABLE_OID value in SYS_TABLES_.

INDEX_ID

This is the identifier of an index currently being replicated by the replication Sender thread.

KEY_COLUMN_ID

This is the identifier of the column on which the index is based.

KEY_COLUMN_FLAG

This is an internal flag for the column on which the index is based.

COMPOSITE_ORDER

This is the position of the column on which the index is based.

Reference Tables

```
SYS_REPL_OLD_ITEMS_
SYS_REPL_OLD_COLUMNS_
SYS_REPL_OLD_INDICES_
```

SYS_REPL_OLD_INDICES_

This meta table contains information about indexes currently being replicated by the replication Sender thread.

Column name	Type	Description
REPLICATION_NAME	VARCHAR(40)	The replication name
TABLE_OID	BIGINT	The object identifier of the table
INDEX_ID	INTEGER	The index identifier
INDEX_NAME	VARCHAR(128)	The index name
TYPE_ID	INTEGER	The index type identifier
IS_UNIQUE	CHAR(1)	Indicates whether or not the index is globally unique
IS_LOCAL_UNIQUE	CHAR(1)	Indicates whether or not the index is locally unique
IS_RANGE	CHAR(1)	Indicates whether or not range scanning is possible using the index

Column Information

REPLICATION_NAME

This is the user-defined replication name. Its value corresponds to a REPLICATION_NAME value in the SYS_REPLICATIONS_ meta table.

TABLE_OID

This is the identifier of a table currently being replicated by the replication Sender thread. Its value may be different from that of TABLE_OID in the SYS_TABLES_ meta table.

INDEX_ID

This is the identifier of an index currently being replicated by the replication Sender thread.

INDEX_NAME

This is the name of an index currently being replicated by the replication Sender thread.

TYPE_ID

This is an index type identifier, and is an internal value.

IS_UNIQUE

This indicates whether or not the index is globally unique. 'Y' signifies that the index is globally unique, and 'N' signifies that it is not globally unique.

IS_LOCAL_UNIQUE

This indicates whether or not the index is locally unique. 'Y' signifies that it is locally unique, and 'N' means that it is not locally unique.

IS_RANGE

This indicates whether or not the index is locally unique. 'Y' signifies that it is locally unique, and 'N' means that it is not locally unique.

Reference Tables

```
SYS_REPL_OLD_ITEMS_
SYS_REPL_OLD_COLUMNS_
SYS_REPL_OLD_INDEX_COLUMNS_
```

SYS_REPL_OLD_ITEMS_

This meta table contains information about tables currently being replicated by the replication Sender thread.

Column name	Type	Description
REPLICATION_NAME	VARCHAR(40)	The partition name
TABLE_OID	BIGINT	The table object identifier
USER_NAME	VARCHAR(128)	The user name
TABLE_NAME	VARCHAR(128)	The table name
PARTITION_NAME	VARCHAR(128)	The name of the replication
PRIMARY_KEY_INDEX_ID	INTEGER	The index identifier of the primary key

Column Information

REPLICATION_NAME

This value corresponds to a REPLICATION_NAME in the SYS_REPLICATIONS_ meta table, and is the user-defined replication name.

TABLE_OID

This is the identifier of a table currently being replicated by the replication Sender thread. Its value may be different from the value of TABLE_OID in the SYS_TABLES_ meta table.

USER_NAME

This is the user name of the owner of the table being replicated on the local server. Its value corresponds to a USER_NAME in the SYS_USERS_ meta table.

TABLE_NAME

This is the name of the table being replicated on the local server. Its value corresponds to a TABLE_NAME value in the SYS_TABLES_ meta table.

PARTITION_NAME

This is the name of the partition containing the table being replicated on the local server.

PRIMARY_KEY_INDEX_ID

This is the identifier of a primary key index.

Reference Tables

```
SYS_REPL_OLD_COLUMNS_
SYS_REPL_OLD_INDICES_
SYS_REPL_OLD_INDEX_COLUMNS_
```

SYS_REPL_TABLE_OID_IN_USE_

This meta table is for managing information about TABLE OID of tables included in DDL log but not yet replicated.

Column name	Type	Description
REPLICATION_NAME	VARCHAR(40)	The name of the replication object
OLD_TABLE_OID	BIGINTBIGINT	The old object identifier of the table before DDL
TABLE_OID	BIGINTBIGINT	The current object identifier of the table

Column Information

REPLICATION_NAME

This is the replication name, which is specified by the user. It corresponds to a REPLICATION_NAME in the SYS_REPLICATIONS_ meta table.

OLD_TABLE_OID

This is the old object identifier of the table that is included in DDL log not yet replicated.

TABLE_OID

This is the current object identifier of the table that is included in DDL log not yet replicated. It corresponds to a TABLE_OID in the SYS_REPL_ITEMS_ meta table.

SYS_REPL_RECOVERY_INFOS_

This is the meta table in which log information is written for use in recovery of the remote server.

Column name	Type	Description
REPLICATION_NAME	VARCHAR(40)	The name of the replication
MASTER_BEGIN_SN	BIGINT	The starting log number of a master transaction
MASTER_COMMIT_SN	BIGINT	The final log number of the master transaction
REPLICATED_BEGIN_SN	BIGINT	The starting log number of a replication transaction
REPLICATED_COMMIT_SN	BIGINT	The final log number of the replication transaction

Column Information

REPLICATION_NAME

This is the replication object name defined by the user, and corresponds to a REPLICATION_NAME in the SYS_REPLICATIONS_ meta table.

MASTER_BEGIN_SN

The starting log number of a master transaction occurring on a remote server.

MASTER_COMMIT_SN

The final log number of a master transaction occurring on a remote server.

REPLICATED_BEGIN_SN

The starting log number of a replication transaction occurring on the local server.

REPLICATED_COMMIT_SN

The final log number of a replication transaction occurring on the local server.

Reference Table

```
SYS_REPLICATIONS_
```

SYS_SECURITY_

This table contains information about the state of the security module.

Column name	Type	Description
MODULE_NAME	VARCHAR(24)	The name of the security module
MODULE_VERSION	VARCHAR(40)	The version of the security module
ECC_POLICY_NAME	VARCHAR(16)	The name of the ECC policy
ECC_POLICY_CODE	VARCHAR(64)	The verification code of the ECC policy

This table shows whether a security module authored by a third party is being used.

In the case where a security module authored by a third party is in use, the SYS_SECURITY_ meta table contains information about the properties of the security module, whereas if no such security module is in use, the SYS_SECURITY_ meta table will contain no records.

SYS_SYNONYMS_

This is the table for storing information about synonyms, which provide alias functions for database objects.

Column name	Type	Description
SYNONYM_OWNER_ID	INTEGER	The user identifier
SYNONYM_NAME	VARCHAR(128)	The synonym name
OBJECT_OWNER_NAME	VARCHAR(128)	The name of the object owner
OBJECT_NAME	VARCHAR(128)	The name of the synonym target object
CREATED	DATE	The time at which the synonym was created
LAST_DDL_TIME	DATE	The most recent time at which a DDL statement was used to make changes to a synonym

Column Information

SYNONYM_OWNER_ID

This is the identifier of the owner of the synonym, and corresponds to a USER_ID in the SYS_USERS_ meta table.

SYNONYM_NAME

This is the synonym name, which is defined by the user.

OBJECT_OWNER_NAME

This is the name of the owner of the schema containing the object that is the target of the user-defined synonym.

OBJECT_NAME

This is the name of the object targeted by the user-defined synonym.

CREATED

This is the time at which the synonym was created.

LAST_DDL_TIME

This is the most recent time at which a DDL statement was used to create or make changes to the synonym.

Reference Table

```
SYS_USERS_
```

SYS_TABLES_

This table contains information about meta tables, user-defined tables, sequences and views.

Column name	Type	Description
USER_ID	INTEGER	The user identifier
TABLE_ID	INTEGER	The table identifier
TABLE_OID	BIGINT	The table object identifier
COLUMN_COUNT	INTEGER	The number of columns in the table
TABLE_NAME	VARCHAR(128)	The name of the table
TABLE_TYPE	CHAR(1)	The object type
REPLICATION_COUNT	INTEGER	The number of replications related to the table

Column name	Type	Description
REPLICATION_RECOVERY_COUNT	INTEGER	The number of replications that use the recovery option and are related to the table
MAXROW	BIGINT	The maximum number of records that can be entered (0: no limit)
TBS_ID	INTEGER	The tablespace identifier
TBS_NAME	VARCHAR(128)	The name of the tablespace in which the table is stored
PCTFREE	INTEGER	See below
PCTUSED	INTEGER	See below
INIT_TRANS	INTEGER	The initial number of transactions that can be simultaneously used for update in a page
MAX_TRANS	INTEGER	The maximum number of transactions that can be simultaneously used for update in a page
INITEXTENTS	BIGINT	The initial number of extents when a table is created
NEXTEXTENTS	BIGINT	The number of extents that are added when a table is expanded
MINEXTENTS	BIGINT	The minimum number of extents in a table
MAXEXTENTS	BIGINT	The maximum number of extents in a table
IS_PARTITIONED	CHAR(1)	Indicates whether a table is partitioned
TEMPORARY	CHAR(1)	Whether or not the table is a temporary table D: Is a transaction-specific temporary table P: Is a session-specific temporary table N: Is not a temporary table
HIDDEN	CHAR(1)	Indicates whether the table has hidden properties
ACCESS	CHAR(1)	The data access mode for the table

Column name	Type	Description
PARALLEL_DEGREE	INTEGER	The number of threads which execute parallel queries
CREATED	DATE	The time at which the table was created
LAST_DDL_TIME	DATE	The time at which the table was most recently changed using a DDL statement

Column Information

USER_ID

This is the identifier of the owner of the table, and corresponds to a USER_ID in the SYS_USERS_ meta table

TABLE_ID

This is the table identifier, which is automatically assigned by the system sequence.

TABLE_OID

This is the table object identifier, which is automatically and internally assigned by the system. Unlike TABLE_ID, which is used when the user reads meta tables, this value is used only for internal operations.

COLUMN_COUNT

This is the number of columns defined in the table.

TABLE_NAME

This is the table name, which is defined by the user.

TABLE_TYPE

Information not only about tables, but also about sequences, views, etc. is saved in the SYS_TABLES_ meta table. This type identifier is used to distinguish them, and consists of the following types:

- T: A table
- S: A sequence
- V: A view
- W: A sequence for queue use only
- Q: A queue
- M: A table automatically created for the maintenance of the data of the materialized view
- A: A view automatically created for the maintenance of the data of the materialized view
- G: An internal table for global index of
- D: A dictionary table internally used for actually storing compressed column data

REPLICATION_COUNT

This is the number of replication objects associated with the table.

REPLICATION_RECOVERY_COUNT

This is the number of replication objects that use the recovery option and are associated with the table.

MAXROW

This is the maximum number of records that can be inserted into the table.

TBS_ID

This is the identifier of the tablespace in which the table is saved.

PCTFREE

This is the minimum percentage of free space that must exist in order for it to be possible to update a page. Usually, an amount of space equal to the percentage specified in PCTFREE is kept free so that existing rows saved in a page can be updated. For example, if PCTFREE is set to 20, 20% of the space in the page is set aside for update operations, so data can be inserted only into 80% of the space in the page.

The user can set PCTFREE between 0 and 99 when executing the CREATE TABLE statement.

PCTUSED

This is a threshold below which the amount of used space in a page must decrease in order for the page to return to the state in which records can be inserted from the state in which only update operations are possible. If the amount of free space falls below the percentage specified in PCTFREE, it will become impossible to insert new records into the page, and it will only be possible to update and delete rows. If subsequent update or delete operations reduce the percentage of used space below the threshold specified by PCTUSED, it will become possible to insert new rows into the page again.

The user can set PCTUSED between 0 and 99 when the CREATE TABLE statement is executed.

> * For more detailed explanations of PCTFREE and PCTUSED, please refer to the description of the CREATE TABLE statement in the *SQL Reference*.

INIT_TRANS

This is the initial number of update transactions that can be simultaneously executed, and is set when a page is created. The actual number of transactions can increase to the number specified in MAX_TRANS, as long as sufficient page space is available.

MAX_TRANS

This is the maximum number of update transactions that can be simultaneously executed for a single page.

INITEXTENTS

This denotes the number of extents that are available to be allocated when a table is created.

NEXTEXTENTS

This denotes the number of additional extents that are available to be allocated when the size of a table is increased.

MINEXTENTS

This denotes the minimum number of available extents for a table.

MAXEXTENTS

This denotes the maximum number of available extents for a table.

IS_PARTITIONED

This is an identifier that indicates whether a table is partitioned. If it is 'Y', the table is partitioned. If it is 'F', the table is not partitioned.

TEMPORARY

This indicates whether or not the table is a temporary table.

- D: A transaction-specific temporary table
- P: A session-specific temporary table
- N: Not a temporary table

HIDDEN

This indicates whether the table is hidden or not.

- Y: The table is hidden from the user
- N: the table is open to the user (normal table)

PARALLEL_DEGREE

This indicates the number of threads which execute parallel queries when scanning the partitioned table.

ACCESS

This is the data access mode for the table. The default mode is W which allows Read/Write.

- R: Read-Only mode
- W: Read/Write mode (default mode)
- A: Read/Add mode. This mode disallows the alteration/deletion of data.

Reference Table

SYS_USERS_

SYS_TABLE_PARTITIONS_

This is a meta table for the management of table partitions.

Column name	Type	Description
USER_ID	INTEGER	The user identifier
TABLE_ID	INTEGER	The table identifier
PARTITION_OID	BIGINT	The partition object identifier
PARTITION_ID	INTEGER	The partition identifier
PARTITION_NAME	VARCHAR(128)	The partition name
PARTITION_MIN_VALUE	VARCHAR(4000)	The minimum reference value for a partition (NULL in the case of a hash partition)
PARTITION_MAX_VALUE	VARCHAR(4000)	The maximum reference value for a partition (NULL in the case of a hash partition)
PARTITION_ORDER	INTEGER	The position of the partition (required for hash partitions)
TBS_ID	INTEGER	The identifier of a tablespace
PARTITION_ACCESS	CHAR(1)	The data access mode for the partition
REPLICATION_COUNT	INTEGER	The number of replication objects related to this partition
REPLICATION_RECOVERY_COUNT	INTEGER	The number of replication objects which have enabled the recovery option for this partition
CREATED	DATE	The date and time at which the parition is created
LAST_DDL_TIME	DATE	The time at which the partition was most recently changed using a DDL statement

Column Information

USER_ID

This is the identifier of the table owner, and corresponds to a USER_ID in the SYS_USERS_ meta table.

TABLE_ID

This is the table identifier. It is assigned automatically by the system sequence.

PARTITION_OID

This is the partition object identifier. It is assigned automatically by the system. Unlike PARTITION_ID, which is used when viewing meta tables, it is used only internally by the system.

PARTITION_ID

This is the partition identifier.

PARTITION_NAME

This is the user-defined partition name.

PARTITION_MIN_VALUE

This is a string that gives the minimum reference value for a partition. It is NULL for hash partitions.

PARTITION_MAX_VALUE

This is a string that gives the maximum reference value for a partition. It is NULL for hash partitions.

PARTITION_ORDER

This is the position of the partition among the partitions. It is required for hash partitions.

TBS_ID

This is the identifier of the tablespace in which the table is stored.

PARTITION_ACCESS

This is the data access mode for the partition. The default mode is W which allows Read/Write.

- R: Read-Only mode
- W: Read/Write mode (default mode)
- A: Read/Append mode. This mode disallows the alteration/deletion of data.

REPLICATION_COUNT

This is the number of replication objects related to this partition.

REPLICATION_RECOVERY_COUNT

This is the number of replication objects which have enabled the recovery option for this partition.

Reference Tables

```
SYS_USERS_
SYS_TABLES_
SYS_PART_TABLES_
```

SYS_TABLE_SIZE_

This table stores information about the actual size of disk and memory tables in the system.

Column name	Type	Description
USER_NAME	VARCHAR(128)	The table owner
TABLE_NAME	VARCHAR(128)	The table name
TBS_NAME	VARCHAR(128)	The name of the tablespace in which the table is saved
MEMORY_SIZE	BIGINT	The memory table size
DISK_SIZE	BIGINT	The disk table size

SYS_TBS_USERS_

This meta table contains information about the relationship between users and user-defined tablespaces.

Column name	Type	Description
TBS_ID	INTEGER	The tablespace identifier
USER_ID	INTEGER	The user identifier
IS_ACCESS	INTEGER	Whether the user is allowed to access the tablespace

Column Information

TBS_ID

This is the tablespace identifier.

USER_ID

This is the identifier of a particular user. It corresponds to a USER_ID in the SYS_USERS_ meta table.

IS_ACCESS

This indicates whether the user is permitted to access the tablespace.

- 0: access not permitted
- 1: access permitted.

Reference Table

SYS_USERS_

SYS_TRIGGERS_

This meta table contains default information about triggers.

Column name	Type	Description
USER_ID	INTEGER	The user identifier
USER_NAME	VARCHAR(128)	The user name
TRIGGER_OID	BIGINT	The trigger identifier
TRIGGER_NAME	VARCHAR(128)	The trigger name
TABLE_ID	INTEGER	The table identifier
IS_ENABLE	INTEGER	Indicates whether the trigger is enabled
EVENT_TIME	INTEGER	Indicates when the trigger fires
EVENT_TYPE	INTEGER	The trigger event type
UPDATE_COLUMN_CNT	INTEGER	The number of columns that can cause a trigger to fire if updated
GRANULARITY	INTEGER	The units in which the trigger is executed
REF_ROW_CNT	INTEGER	The number of ALIASes for a REFERENCING statement
SUBSTRING_CNT	INTEGER	The number of records in which the trigger statement is saved
STRING_LENGTH	INTEGER	The total length of the trigger statement character string
CREATED	DATE	The time at which the trigger was created
LAST_DDL_TIME	DATE	The most recent time at which a DDL statement was used to make changes to the trigger

Column Information

USER_ID

This is the identifier of the user who owns the trigger, and corresponds to a USER_ID in the SYS_USERS_ meta table.

USER_NAME

This is the user name, and corresponds to a USER_NAME in the SYS_USERS_ meta table.

TRIGGER_OID

This is the trigger identifier. It is automatically assigned by the system.

TRIGGER_NAME

This is the user-defined trigger name.

TABLE_ID

This is the identifier of the table on which the trigger is defined, and corresponds to a TABLE_ID in the SYS_TABLES_ meta table.

IS_ENABLE

This value indicates whether or not the trigger is enabled. It can be modified using the ALTER TRIGGER statement.

- 0: DISABLED
- 1: ENABLED

EVENT_TIME

This displays the time point that the trigger fires.

- 1: BEFORE
- 2: AFTER
- 3: INSTEAD OF

EVENT_TYPE

This is the type of the event that causes the trigger to fire.

- 1: INSERT
- 2: DELETE
- 4: UPDATE

UPDATE_COLUMN_CNT

This is the number of columns that cause a trigger to fire when updated. This value is equal to the number of records related to the trigger in the SYS_TRIGGER_UPDATE_COLUMNS_ meta table.

GRANULARITY

This value indicates how often the trigger fires:

- 1: FOR EACH ROW
- 2: FOR EACH STATEMENT

REF_ROW_CNT

This is the number of ALIASes defined in a REFERENCING statement.

SUBSTRING_CNT

One trigger statement is divided into several records and stored in the SYS_TRIGGER_STRINGS_ meta table. This value indicates the number of records used to store the statement.

STRING_LENGTH

This is the total length of the trigger statement character string.

Reference Tables

```
SYS_USERS_
SYS_TABLES_
```

SYS_TRIGGER_DML_TABLES_

This meta table contains information about tables referenced by triggers.

Column name	Type	Description
TABLE_ID	INTEGER	The table identifier
TRIGGER_OID	BIGINT	The trigger identifier
DML_TABLE_ID	INTEGER	The table identifier within the trigger
STMT_TYPE	INTEGER	The type of executable statement

Column Information

TABLE_ID

This is the identifier of the table on which the trigger is defined, and corresponds to a TABLE_ID in the SYS_TABLES_ meta table.

TRIGGER_OID

This is the trigger identifier, and corresponds to a TRIGGER_OID in the SYS_TRIGGERS_ meta table.

DML_TABLE_ID

This is the identifier of the table that is accessed using the DML statements within the trigger, and corresponds to a TABLE_ID in the SYS_TABLES_ meta table

STMT_TYPE

This is the type of statement executed on a table.

- 8: DELETE
- 19: INSERT
- 33: UPDATE

Reference Tables

```
SYS_TABLES_
SYS_TRIGGERS_
```

SYS_TRIGGER_STRINGS_

This is the meta table in which the trigger statements are saved.

Column name	Type	Description
TABLE_ID	INTEGER	The table identifier
TRIGGER_OID	BIGINT	The trigger identifier
SEQNO	INTEGER	The position of this text fragment in the trigger statement
SUBSTRING	VARCHAR(100)	A fragment of trigger statement text

Column Information

TABLE_ID

This is the table identifier, and corresponds to a TABLE_ID in the SYS_TABLES_ meta table.

TRIGGER_OID

This is the trigger identifier, and corresponds to a TRIGGER_OID in the SYS_TRIGGERS_ meta table.

SEQNO

When information about a single trigger statement is saved as several records in SYS_TRIGGER_STRINGS, this is the position of this record among the records.

SUBSTRING

This is a fragment of the trigger statement text. When records are searched for using a single TRIGGER_OID and their SUBSTRING values are concatenated in the order described in SEQNO, the complete trigger command can be reconstructed.

SYS_TRIGGER_UPDATE_COLUMNS_

This meta table contains information about columns that cause triggers to fire when updated.

Column name	Type	Description
TABLE_ID	INTEGER	The table identifier
TRIGGER_OID	BIGINT	The trigger identifier
COLUMN_ID	INTEGER	The column identifier

Column Information

TABLE_ID

This is the table identifier, and corresponds to a TABLE_ID in the SYS_TABLES_ meta table.

TRIGGER_OID

This is the trigger identifier, and corresponds to a TRIGGER_OID in the SYS_TRIGGERS_ meta table.

COLUMN_ID

This is the column ID, and corresponds to a COLUMN_ID in the SYS_COLUMNS_ meta table.

Reference Tables

```
SYS_TABLES_
SYS_TRIGGERS_
```

SYS_USERS_

This meta table contains information about database users.

Column name	Type	Description
USER_ID	INTEGER	The user identifier
USER_NAME	VARCHAR(128)	The user name
PASSWORD	VARCHAR(256)	The user password
DEFAULT_TBS_ID	INTEGER	The default tablespace identifier
TEMP_TBS_ID	INTEGER	The temporary tablespace identifier

Column name	Type	Description
ACCOUNT_LOCK	CHAR(1)	Whether the account is locked/unlocked N: UNLOCKED L: LOCKED
ACCOUNT_LOCK_DATE	DATE	The date on which the account was locked
PASSWORD_LIMIT_FLAG	CHAR(1)	Indicates the use of password management policies T: used F: not used
FAILED_LOGIN_ATTEMPTS	INTEGER	The maximum number of times login failure is permitted
FAILED_LOGIN_COUNT	INTEGER	The number of times login fails
PASSWORD_LOCK_TIME	INTEGER	The amount of time needed to elapse for a locked account to become unlocked
PASSWORD_EXPIRY_DATE	DATE	The password expiry date
PASSWORD_LIFE_TIME	INTEGER	The password validity period
PASSWORD_GRACE_TIME	INTEGER	The password grace period following expiration
PASSWORD_REUSE_DATE	DATE	The date on which identical passwords are available for reuse
PASSWORD_REUSE_TIME	INTEGER	Not used
PASSWORD_REUSE_MAX	INTEGER	The number of times identical passwords are available for reuse
PASSWORD_REUSE_COUNT	INTEGER	Not used
PASSWORD_VERIFY_FUNCTION	VARCHAR(128)	The CALLBACK function for verifying passwords
USER_TYPE	CHAR(1)	Indicates the user type U: User R: Role
DISABLE_TCP	CHAR(1)	Availability of TCP connection T: TCP connection is disabled; SSL or IPC is enabled. F: TCP connection is enabled.
CREATED	DATE	The time at which the database user was created
LAST_DDL_TIME	DATE	The most recent time at which a DDL statement was used to make changes to the user

Column Information

USER_ID

This is the user identifier. It is automatically assigned by the system sequence.

USER_NAME

This is the user-defined user name.

PASSWORD

This is the encrypted user password.

DEFAULT_TBS_ID

This is the identifier of the default tablespace, which is used when the user creates an object without explicitly specifying a tablespace.

TEMP_TBS_ID

This is the identifier for the user temporary tablespace.

DISABLE_TCP

Displays the availability of TCP connection.

Reference Table

DBA_USERS_

DBA_USERS_

This is a meta table which records the information of database user and it can only be viewed by the SYS user. Note that the information of tables and columns is identical with the SYS_USERS_.

Column name	Type	Description
USER_ID	INTEGER	The user identifier
USER_NAME	VARCHAR(128)	The user name
PASSWORD	VARCHAR(256)	The user password
DEFAULT_TBS_ID	INTEGER	The default tablespace identifier
TEMP_TBS_ID	INTEGER	The temporary tablespace identifier
ACCOUNT_LOCK	CHAR(1)	Indicates whether account is locked N: UNLOCKED L: LOCKED
ACCOUNT_LOCK_DATE	DATE	The data the account was locked

Column name	Type	Description
PASSWORD_LIMIT_FLAG	CHAR(1)	Indicates whether password management policy is used. T: Enable password management policy F: Disable password management policy
FAILED_LOGIN_ATTEMPTS	INTEGER	The maximum number of failed login attempts
FAILED_LOGIN_COUNT	INTEGER	The login failed count
PASSWORD_LOCK_TIME	INTEGER	The amount of time taht must elapse after an account is locked once and then released again
PASSWORD_EXPIRY_DATE	DATE	The password expiration date
PASSWORD_LIFE_TIME	INTEGER	The passwrod expiration time
PASSWORD_GRACE_TIME	INTEGER	The grace time after password expiration
PASSWORD_REUSE_DATE	DATE	The date when the same password will be reused
PASSWORD_REUSE_TIME	INTEGER	Not used
PASSWORD_REUSE_MAX	INTEGER	The number of reuse of the same password
PASSWORD_REUSE_COUNT	INTEGER	Not used
PASSWORD_VERIFY_FUNCTION	VARCHAR(128)	The Callback function to verify password
USER_TYPE	CHAR(1)	The user type display U: User R: Role
DISABLE_TCP	CHAR(1)	Indicates whether or not TCP connection is in use. T: Failed TCP connection, communication only with SSL or IPC F: Allow TCP connection
CREATED	DATE	The time when the database user was created
LAST_DDL_TIME	DATE	The time when the last DDL change occurred for a user

Column Information

USER_ID

This is the user identifier, automatically assigned by a sequence in the system.

USER_NAME

This is the name of the user specified by the user.

PASSWORD

This is encrypted with the user's password.

DEFAULT_TBS_ID

This is the default tablespace identifier, used when the user does not explicitly specify a tablespace when creating an object.

TEMP_TBS_ID

This is the user's temporary tablespace identifier.

DISABLE_TCP

This indicates to allow or restrict the user's TCP connection.

SYS_USER_ROLES_

This meta table stores information about the roles granted to the user.

Column name	Type	Description
GRANTOR_ID	INTEGER	The identifier of the user to whom the role is granted
GRANTEE_ID	INTEGER	The identifier of the user who granted the role
ROLE_ID	INTEGER	The role identifier

Column Information

GRANTOR_ID

This is the identifier of the user who granted the role, and corresponds to a USER_ID value in the SYS_USERS_ meta table.

GRANTEE_ID

This is the identifier of the user to whom the role is granted, and corresponds to a USER_ID value in the SYS_USERS_ meta table.

ROLE_ID

This is the role identifier, and corresponds to a USER_ID value in the SYS_USERS_ meta table.

Reference Table

```
SYS_USERS_
SYS_TABLES_
```

SYS_VIEWS_

Basic information about views is stored in the SYS_TABLES_ meta table. This meta table contains additional information about views.

Column name	Type	Description
USER_ID	INTEGER	The identifier of the owner of the view
VIEW_ID	INTEGER	The view identifier
STATUS	INTEGER	The view status
READ_ONLY	CHAR(1)	Displays whether the view is a read-only view

Column Information

USER_ID

This is the identifier of the view owner, and corresponds to a USER_ID in the SYS_USERS_ meta table.

VIEW_ID

This is the view identifier, and corresponds to a TABLE_ID in the SYS_TABLES_ meta table.

STATUS

This value indicates the status of the view:

- 0: VALID
- 1: INVALID

Reference Tables

```
SYS_USERS_
SYS_TABLES_
```

SYS_VIEW_PARSE_

This meta table contains the text of view creation statements.

Column name	Type	Description
USER_ID	INTEGER	The identifier of the owner of the view
VIEW_ID	INTEGER	The identifier of the view
SEQ_NO	INTEGER	When a view creation statement text is split and the text is saved as multiple text fragments in SYS_VIEW_PARSE_, this is the position of the record among the records.
PARSE	VARCHAR(100)	A text fragment of the view creation statement

Column Information

USER_ID

This is the identifier of the view owner, and corresponds to a USER_ID in the SYS_USERS_ meta table.

VIEW_ID

This is the view identifier, and corresponds to a TABLE_ID in the SYS_TABLES_ meta table.

SEQ_NO

When a single statement corresponding to one view is saved as multiple records in SYS_VIEW_PARSE_, this is the position of the record among the records.

PARSE

When records are searched for using a single VIEW_ID and their PARSE values are concatenated in the order described in SEQ_NO, the complete view statement can be reconstructed.

Reference Tables

```
SYS_USERS_
SYS_TABLES_
```

SYS_VIEW_RELATED_

This meta table contains information about objects accessed by user-defined views.

Column name	Type	Description
USER_ID	INTEGER	The identifier of the owner of the view
VIEW_ID	INTEGER	The view identifier
RELATED_USER_ID	INTEGER	The identifier of the owner of the object that the view accesses
RELATED_OBJECT_NAME	VARCHAR(128)	The name of the object accessed by the view

Column name	Type	Description
RELATED_OBJECT_TYPE	INTEGER	The type of the object accessed by the view

Column Information

USER_ID

This is the identifier of the view owner, and corresponds to a USER_ID in the SYS_USERS_ meta table.

VIEW_ID

This is the identifier of the view, and corresponds to a TABLE_ID in the SYS_TABLES_ meta table.

RELATED_USER_ID

This is the identifier of the owner of the object accessed by the view, and corresponds to a USER_ID in the SYS_USERS_ meta table.

RELATED_OBJECT_NAME

This is the name of the object accessed by the view.

RELATED_OBJECT_TYPE

This identifies the type of object accessed by the view. Views can access stored functions, tables, sequences, other views, Database Link objects, and synonyms. The identifiers are as follows:

- 1: Stored function
- 2: Table, Sequence, View
- 4: Database link
- 5: Synonym

Reference Tables

```
SYS_USERS_
SYS_TABLES_
SYS_PROCEDURES_
```

SYS_XA_HEURISTIC_TRANS_

This is a meta table that contains identifiers and information about the status of the database's global transactions.

Column name	Type	Description
FORMAT_ID	BIGINT	The identifier of the format of the global transaction
GLOBAL_TX_ID	VARCHAR(128)	The identifier of the global transaction
BRANCH_QUALIFIER	VARCHAR(128)	The branch qualifier of the global transaction

Column name	Type	Description
STATUS	INTEGER	The status of the global transaction
OCCUR_TIME	DATE	The time XA transaction occurred.

Column Information

FORMAT_ID

This is the identifier of the format of the global transaction.

GLOBAL_TX_ID

This is the identifier of the global transaction.

BRANCH_QUALIFIER

This is the branch qualifier of the global transaction.

STATUS

This is the status of the global transaction.

SYS_GEOMETRIES_

This is a meta table that contains information about tables that have GEOMETRY columns.

Column name	Type	Description
USER_ID	INTERGER	The identifier of the user
TABLE_ID	INTERGER	The table identifier
COLUMN_ID	INTERGER	The column identifier
COORD_DIMENSION	INTERGER	The dimension of the GEOMETRY object
SRID	INTERGER	The spatial reference identifier in the database

SYS_GEOMETRY_COLUMNS_

This meta table is used to manage and specify SRID in the GEOMETRY column. The synonym of this meta tble is GEOMETRY_COLUMNS_.

Column name	Type	Description
F_TABLE_SCHEMA	VARCHAR(128)	The name of the owner of the table
F_TABLE_NAME	VARCHAR(128)	The name of the table
F_GEOMETRY_COLUMN	VARCHAR(128)	The name of the column
COORD_DIMENSION	INTERGER	The dimension of the GEOMETRY object

Column name	Type	Description
SRID	INTERGER	The spatial reference identifier in the database

USER_SRS_

This meta table is used to manage information about SRID and the SRS according to it. The synonym of this meta table is SPATIAL_REF_SYS.

To add SRS meta data to SPATIAL_REF_SYS table and to delete from it, ADD_SPATIAL_REF_SYS and DELETE_SPATIAL_REF_SYS procedures in SYS_SPATIAL package should be used. It is recommended to set SRID and AUTH_SRID's value the same when adding the meta data. For more information, please refer to *Spatial Manual*.

Column name	Type	Description
SRID	INTEGER	The spatial reference identifier in the database
AUTH_NAME	VARCHAR(256)	The standard name
AUTH_SRID	INTEGER	The standard Spatial Reference Identifier
SRTEXT	VARCHAR(2048)	The description of the Spatial Reference System in OGC-WKT form
PROJ4TEXT	VARCHAR(2048)	The information for used in PROJ4

Performance Views

Performance views are structures that exist in memory but have the form of regular tables, and allow users to monitor internal information about an Altibase system, such as system memory, process status, sessions, buffers, threads, etc.

Performance views allow Altibase users to easily obtain information about memory objects (e.g. session information, log information, thread information) using SQL statements while Altibase is running, in the same way that they would use SQL to search for data saved in regular tables.

This section describes the kinds of performance views provided with Altibase, their structure and function, how to access them, and the information that each view provides.

Structures and Features

Inside Altibase there is not only information about user-created objects such as tables; there is also a variety of information required for the operation of the DBMS itself. Because Altibase has a hybrid structure, in which tables can be created and queried not only in memory space but also in disk space, monitoring Altibase is particularly critical.

Performance views provide information about most of the internal memory structures used by Altibase processes in the form of views. Because the data is dynamically created in real time when a view is queried, users can always obtain up-to-date information about internal processes.

Performance views are always read-only. If a user attempts to modify the data in a performance view, Altibase returns an error and rolls back the transaction.

How to Use Performance Views

The entire list of performance views can be retrieved in iSQL as follows:

```
iSQL> SELECT * FROM V$TAB;
```

Performance view schemas can be checked from iSQL using the DESC command, just as with regular tables, and SELECT statements can also be used to query data in the same way that they would be used to query regular tables

V$Views

Performance views are identified by the prefix V$. The following table lists all performance views.

Name	Description
V$ACCESS_LIST	Information about access or block on specific IP packets
V$ALLCOLUMN	Information about the columns that make up a performance view
V$ARCHIVE	Archive and backup- related information
V$BACKUP_INFO	Information about incremental backups performed until now
V$BUFFPAGEINFO	Statistics on the buffer frame of the buffer manager
V$BUFFPOOL_STAT	Buffer pool related statistics, including the buffer pool hit ratio
V$CATALOG	Information about the structure of tables
V$DATABASE	Internal information about memory database space
V$DATAFILES	Information about data files which are related to tablespaces
V$DATATYPE	Information about data types supported by Altibase
V$DBA_2PC_PENDING	A list of distributed transactions whose status is "in-doubt"
V$DBLINK_ALTILINKER_STATUS	Status information about the AltiLinker process for the database link

Name	Description
V$DBLINK_DATABASE_LINK_INFO	Information about the database link object existing in the database
V$DBLINK_GLOBAL_TRANSACTION_INFO	Information about the transactions using the database link
V$DBLINK_LINKER_CONTROL_SESSION_INFO	Status information about the Linker Control Session
V$DBLINK_LINKER_DATA_SESSION_INFO	Status information about the Linker Data Sessions
V$DBLINK_LINKER_SESSION_INFO	Information about the number of Linker Control Session and Linker Data Sessions.
V$DBLINK_NOTIFIER_TRANSACTION_INFO	Information about distributed transaction (on which a failure occurred) AltiLinker processes.
V$DBLINK_REMOTE_STATEMENT_INFO	Information about statements that are executed on the remote server when using Database Link
V$DBLINK_REMOTE_TRANSACTION_INFO	Information about transactions that occur on the remote server when using Database Link
V$DBMS_STATS	Statistical information about the whole database
V$DB_FREEPAGELISTS	Information about all usable page lists
V$DB_PROTOCOL	Information about database protocols input into the server
V$DIRECT_PATH_INSERT	Information about historical statistics on direct-path uploads
V$DISKTBL_INFO	Information about disk tables
V$DISK_BTREE_HEADER	Information about headers of disk BTREE indexes
V$DISK_RTREE_HEADER	Information about headers of disk RTREE indexes
V$DISK_TEMP_INFO	Information about the minimum value of memory to line up the disk temporary tables
V$DISK_TEMP_STAT	Information on each disk temporary table

Name	Description
V$DISK_UNDO_USAGE	Information about the amount of undo tablespace on disk that is currently being used
V$EVENT_NAME	Information about Altibase server wait events
V$EXTPROC_AGENT	Information about the Agent Process created for the execution of external procedures
V$FILESTAT	Statistical information about disk data file I/O
V$FLUSHER	Information about the flusher which flushes the buffers
V$FLUSHINFO	Buffer flush information
V$INDEX	Information about table indexes
V$INSTANCE	Information about the current startup phase
V$INTERNAL_SESSION	Information about a session created in the DBMS_CONCURRENT_EXEC package
V$LATCH	Information about the Buffer Control Block (BCB) latch of the buffer pool and statistical information about read/write latch attempts made on data pages
V$LFG	Information about LFG and statistical information related to GROUP COMMIT
V$LOCK	Information about all table level lock nodes in the database at the current point in time
V$LOCK_STATEMENT	Information about locks and statements, shown together
V$LOCK_WAIT	Information about the status of transactions waiting to obtain locks
V$LOG	Information about log anchor files
V$MEMGC	Information about garbage collection (memory space recovery)
V$MEMSTAT	Statistical information about memory use by Altibase processes
V$MEMTBL_INFO	Information about memory tables
V$MEM_BTREE_HEADER	Information about headers of memory BTREE indexes

Name	Description
V$MEM_BTREE_NODEPOOL	Information about node pools for memory BTREE Indices
V$MEM_RTREE_HEADER	Information about headers of memory RTREE indexes
V$MEM_RTREE_NODEPOOL	Information about node pools for memory RTREE indexes
V$MEM_TABLESPACES	Information about tablespaces created in memory
V$MEM_TABLESPACE_CHECKPOINT_PATHS	Information about the location of DB files in which to record checkpointing details during checkpointing
V$MEM_TABLESPACE_STATUS_DESC	Internal information about the status of memory tablespaces
V$MUTEX	Statistical information about mutexes, used by Altibase for concurrency control
V$NLS_PARAMETERS	Information about parameters related to NLS
V$NLS_TERRITORY	Information about the names of territories available to be set for the database or the current session
V$OBSOLETE_BACKUP_INFO	Backup information no longer required to be retained
V$PKGTEXT	Information about strings of the packages executed on the system
V$PLANTEXT	Information about SQL execution plan text
V$PROCTEXT	Information about stored procedure text
V$PROPERTY	Information about internally set Altibase properties
V$REPEXEC	Information about the replication manager
V$REPGAP	Information about the difference between the log record currently being processed by the replication Sender and the most recently created log record

Name	Description
V$REPGAP_PARALLEL	Information about the difference between the sequence number of the log record currently being processed by replication sender threads working in parallel and the sequence number of the most recently created log record
V$REPLOGBUFFER	Information about the log buffer used for replication
V$REPOFFLINE_STATUS	Information about the status of offline replication execution
V$REPRECEIVER	Information about the replication Receiver
V$REPRECEIVER_COLUMN	Information about target columns for the replication Receiver
V$ REPRECEIVER_PARALLEL	Information about replication Receiver threads working in parallel
V$REPRECEIVER_PARALLEL_APPLY	Information about replication applier threads
V$REPRECEIVER_STATISTICS	Statistical information about the execution time per task of the replication receive thread
V$REPRECEIVER_TRANSTBL	Transaction table information for replication receivers
V$REPRECEIVER_TRANSTBL_PARALLEL	information about transaction tables used by multiple replication Receiver threads working in parallel.
V$REPRECOVERY	Recovery information used in replication
V$REPSENDER	Information about the replication Sender
V$REPSENDER_PARALLEL	Information about replication Sender threads working in parallel
V$REPSENDER_SENT_LOG_COUNT	Information about the number of logs sent by the replication Sender for each DML type
V$REPSENDER_SENT_LOG_COUNT_PARALLEL	Information about the number of logs sent by each Sender thread for each DML type in parallel replication in EAGER mode
V$REPSENDER_STATISTICS	Statistical information about the execution time for each task of the replication send thread

Name	Description
V$REPSENDER_TRANSTBL	Information about transaction tables used by the replication Sender
V$REPSENDER_TRANSTBL_PARALLEL	Information about transaction tables used by replication Sender threads working in parallel
V$REPSYNC	Information about tables that are synchronized using replication
V$SBUFFER_STAT	Statistical information about secondary buffers
V$SEGMENT	Information about segments, which constitute tables and indexes
V$SEQ	Sequence-related information
V$SERVICE_THREAD	Information about service threads related to multiplexing
V$SERVICE_THREAD_MGR	Dynamic Service threads status information related to multiplexing.
V$SESSION	Information about sessions created internally in Altibase
V$SESSION_EVENT	Statistical information about all wait events for all currently connected sessions
V$SESSION_WAIT	Information about wait events for all currently connected sessions
V$SESSION_WAIT_CLASS	Cumulative wait statistic information classified by session, wait event and wait class for all currently connected sessions
V$SESSIONMGR	Statistical information about Altibase sessions
V$SESSTAT	Information about the status of currently connected sessions
V$SFLUSHER	Information about tasks flushing secondary buffer pages to disk
V$SFLUSHINFO	Flushing information about secondary buffers
V$SNAPSHOT	The information of SNAPSHOT settings, memory, and disk undo tablespace
V$SQLTEXT	Information about the text of all SQL statements executed in the system

Name	Description
V$SQL_PLAN_CACHE	Information about the current status and statistical information about the SQL Plan Cache
V$SQL_PLAN_CACHE_PCO	Information about Plan Cache objects registered in the SQL Plan Cache
V$SQL_PLAN_CACHE_SQLTEXT	Information about SQL statements registered in the SQL Plan Cach
V$STABLE_MEM_DATAFILES	Information about the paths of data file(s)
V$STATEMENT	Information about statements for all current Altibase sessions
V$STATNAME	Information about the name and status of the system and sessions
V$ST_ANGULAR_UNIT	Reserved for future use
V$ST_AREA_UNIT	Reserved for future use
V$ST_LINEAR_UNIT	Reserved for future use
V$SYSSTAT	Information about the status of the system
V$SYSTEM_CONFLICT_PAGE	Information about latch contention according to page type
V$SYSTEM_EVENT	Cumulative statistical information about waits from startup to the current time, classified according to wait event
V$SYSTEM_WAIT_CLASS	Cumulative statistical information about waits from startup to the current time, classified according to wait class
V$TABLE	Information about records and columns for all performance views
V$TABLESPACES	Information about tablespaces
V$TIME_ZONE_NAMES	Region names, abbreviations and UTC offset values available to be set for the TIME_ZONE property
V$TRACELOG	Information about trace logging
V$TRANSACTION	Information about transaction objects

Name	Description
V$TRANSACTION_MGR	R Information about the transaction manager of Altibase
V$TSSEGS	Information about all TSS segments
V$TXSEGS	Information about bound transaction segments
V$UDSEGS	Information about all undo segments
V$UNDO_BUFF_STAT	Statistical Information about the undo tablespace buffer pool
V$USAGE	Statistical information about the amount of space used by tables and indexes
V$VERSION	Altibase product version information
V$VOL_TABLESPACES	Information about volatile tablespaces
V$WAIT_CLASS_NAME	Information for grouping wait events into classes
V$XID	List of XIDs, which are branches of distributed transactions, that currently exist in the DBMS

V$ACCESS_LIST

This view displays access permission or deny information on specific IP packets accessing to Altibase.

Column name	Type	Description
ID	INTEGER	ACCESS LIST Identifier
ADDRESS	VARCHAR(40)	IP address
OPERATION	VARCHAR(6)	Access permit or deny status of IP address
MASK	VARCHAR(16)	Subnet Mask (IPv4) or prefix big length (IPv6)
LIMIT	INTEGER	Maximum number of sessions allowed
CONNECTED	INTEGER	Number of sessions connected

Column Information

ID

ID describes an identifier for permit or deny list of IP packets.

ADDRESS

ADDRESS describes the IP packet address.

OPERATION

OPERATION displays the status of permit or deny of the IP packet address.

- PERMIT : Access permit
- DENY: Access deny

MASK

If the specified address is in IPv4 address notation, subnet mask is described whereas the length of prefix bit is described if the specified address is in IPv6 address notation. Refer to the description delineated in the ACCESS_LIST property.

LIMIT

Maximum number of sessions allowed to connect to the Altibase server within the IP address range specified in ACCESS_LIST.

If new ACCESS_LIST is added using RELOAD ACCESS LIST while running, the session that is currently connected will not be affected. The updated ACCESS_LIST will only be applied to the new connection requests. For example, when the user specifies the value of LIMIT of ACCESS_LIST and executes RELOAD ACCESS LIST, the value of LIMIT is only applied to the connections created after the change is made. In this case, when V$ACCESS_LIST is inquired, the value of CONNECTED can be bigger than the value of LIMIT.

CONNECTED

Number of sessions that is currently connected to Altibase server within the IP address range specified in ACCESS_LIST.

V$ALLCOLUMN

This view displays information about the columns in all performance views.

Column name	Type	Description
TABLENAME	VARCHAR(39)	The name of the performance view
COLNAME	VARCHAR(39)	The name of the column in the performance view

Column Information

TABLENAME

This is the name of the performance view.

COLNAME

This is the name of the column in the performance view.

V$ARCHIVE

This view displays the information related to archiving and backups.

Column name	Type	Description
LFG_ID	INTEGER	The log file group identifier
ARCHIVE_MODE	BIGINT	Archive log mode 0: no archive log mode 1: archive log mode
ARCHIVE_THR_RUNNING	BIGINT	Information about the execution of the archivelog thread
ARCHIVE_DEST	VARCHAR(1024)	The directory in which logs are to be archived
NEXTLOGFILE_TO_ARCH	INTEGER	The number of the next log file to be archived
OLDEST_ACTIVE_LOGFILE	INTEGER	The number of the oldest of the online log files
CURRENT_LOGFILE	INTEGER	The number of the current online log file

Column Information

LFG_ID

This is the identifier of the LFG which default value is '0'.

ARCHIVE_MODE

This indicates the archive log mode of the database.

0: No archive log mode

1: Archive log mode

V$BACKUP_INFO

This view displays information about all incremenetal backups performed until now.

Column name	Type	Description
BEGIN_BACKUP_TIME	CHAR(24)	The start time of the backup
END_BACKUP_TIME	CHAR(24)	The completion time of the backup
INCREMENTAL_BACKUP_CHUNK_COUNT	INTEGER	The incremental chunk size

Column name	Type	Description
BACKUP_TARGET	INTEGER	The backup target
BACKUP_LEVEL	INTEGER	The backup level
BACKUP_TYPE	INTEGER	The backup type
TABLESPACE_ID	INTEGER	The backup target tablespace ID
FILE_ID	INTEGER	The backup target datafile ID
BACKUP_TAG	CHAR(128)	The backup tag name
BACKUP_FILE	CHAR(512)	The backup file

Column Information

BEGIN_BACKUP_TIME

This indicates the point in time at which backup started and is expressed in the 'YYYY-MM-DD HH:MM:SS' format.

END_BACKUP_TIME

This indicates the point in time at which backup completed and is expressed in the 'YYYY-MM-DD HH:MM:SS' format.

INCREMENTAL_BACKUP_CHUNK_COUNT

0 is always displayed for level 0 incremental backups. The size of an incremental chunk is displayed for level 1 incremental backups.

For more detailed information about incremental chunks, please refer to the INCREMENTAL_BACKUP_CHUNK_SIZE property.

BACKUP_TARGET

This indicates the backup target.

1: Database

2: Tablespace

BACKUP_LEVEL

This indicates the backup level.

1: Level 0

2: Level 1

BACKUP_TYPE

This indicates the backup type.

1: Full backup

2: Differential incremental backup

4: Cumulative incremental backup

TABLESPACE_ID

This indicates the ID of the tablespace to which the backed up datafile belongs.

FILE_ID

This indicates the ID of the backed up datafile.

BACKUP_TAG

This indicates the backup tag name used for the incremental backup.

BACKUP_FILE

This indicates the full path, including the backup file name.

V$BUFFPAGEINFO

This view shows statistics about the main operations managed by the buffer manager for each type of page in the buffer frame.

Column name	Type	Description
PAGE_TYPE	VARCHAR(21)	The type of page
READ_PAGE_COUNT	BIGINT	The number of times that disk I/O (READ) was initiated
GET_PAGE_COUNT	BIGINT	The number of times that buffer frames have been requested
FIX_PAGE_COUNT	BIGINT	The number of times that buffer frames have been fixed
CREATE_PAGE_COUNT	BIGINT	The number of times that new buffer frames have been requested
HIT_RATIO	DOUBLE	The buffer frame hit ratio

Column Information

PAGE_TYPE

PAGE_TYPE indicates the type of buffer page. The possible values are as follows:

PAGE_TYPE	Description
PAGE UNFORMAT	An unformatted page
PAGE FORMAT	A formatted page
PAGE INDEX META BTREE	A page in which meta information about a B-Tree index is written
PAGE INDEX META RTREE	A page in which meta information about an R-Tree index is written
PAGE INDEX BTREE	A page in which a B-Tree index node is written
PAGE INDEX RTREE	A page in which an R-Tree index node is written
PAGE TABLE	A page in which table records are written
PAGE TEMP TABLE META	A page in which meta information about a single temporary table is written
PAGE TEMP TABLE DATA	A page in which the records stored in a temporary table are written
PAGE TSS	A page in which information about the status of a transaction is written. Multiple transaction status slots (TSS) can be written to a single page.
PAGE UNDO	A page in which UNDO information is written. A single page can contain multiple UNDO records.
PAGE LOB DATA	A page in which LOB type data are written. A single page cannot contain more than one LOB column. Moreover, a single LOB column can span multiple pages.
PAGE LOB INODE	A page in which an index node, which pertains to LOB data that exceed a certain size, is written

PAGE_TYPE	Description
PAGE FMS SEGHDR	A page in which a single FMS header is written
PAGE FMS EXTDIR	a pages in which a FMS extent directory is written
PAGE TMS SEGHDR	A page in which a single TMS header is written
PAGE TMS LFBMP	A page in which a single TMS leaf bitmap node is written
PAGE TMS ITBMP	A page in which a single TMS internal bitmap node is written
PAGE TMS RTBMP	A page in which a single TMS root bitmap node is written
PAGE TMS EXTDIR	A page in which a single TMS extent directory is written
PAGE CMS SEGHDR	A page in which a single CMS header is written
PAGE CMS EXTDIR	A page in which a single CMS extent directory is written
PAGE FEBT FSB	A page in which a single datafile header is written
PAGE FEBT EGHA	A page in which meta information about a LOB data column is written
PAGE LOB META	A page in which meta information about a LOB data column is written
PAGE HV TEMP NODE	A page in which a node of a Hash Value-Based Temp Index is written

READ_PAGE_COUNT

This is the total number of disk I/O (read) requests that have been made for buffer frames related to this PAGE_TYPE since the server was started. The value can be 0 or greater

GET_PAGE_COUNT

Shows the total number of read or write requests that have been made to the buffer manager for buffer frames related to this PAGE_TYPE since the server was started. The value can be 0 or greater.

FIX_PAGE_COUNT

This shows the total number of fixes for buffer frames related to PAGE_TYPE received by the buffer manager for reading or writing data since the server was started. The value can be 0 or greater.

CREATE_PAGE_COUNT

This shows the number of requests for new buffer frames for this PAGE_TYPE made to the buffer manager since the server was started. The value can be 0 or greater.

HIT_RATIO

This shows the hit ratio for this buffer since the server was started. Its value can be calculated as follows: (GET_PAGE_COUNT + FIX_PAGE_COUNT - READ_PAGE_COUNT) / (GET_PAGE_COUNT + FIX_PAGE_COUNT)

Example

After the server starts, check the cumulative value of major operations for each page type managed in the buffer.

```
iSQL> select * from v$buffpageinfo;
PAGE_TYPE               READ_PAGE_COUNT      GET_PAGE_COUNT
-------------------------------------------------------------------
FIX_PAGE_COUNT          CREATE_PAGE_COUNT    HIT_RATIO
-------------------------------------------------------------------
PAGE UNFORMAT             0                     0
0                       0                     0
PAGE FORMAT               0                     0
0                       0                     0
PAGE INDEX META BTREE   4                       0
4                       0                     0
PAGE INDEX META RTREE   0                       0
0                       0                     0
PAGE INDEX BTREE          12                    0
12                      0                     0
PAGE INDEX RTREE          0                     0
0                       0                     0
PAGE TABLE                0                     0
0                       0                     0
PAGE TEMP TABLE META     0                      0
0                       0                     0
PAGE TEMP TABLE DATA     0                      0
0                       0                     0
PAGE TSS                  0                     0
0                       0                     0
PAGE UNDO                 0                     0
0                       0                     0
```

```
PAGE LOB DATA            0                         0
0                   0                         0
PAGE LOB INODE           0                         0
0                   0                         0
PAGE FMS SEGHDR          0                         0
0                   0                         0
PAGE FMS EXTDIR          0                         0
0                   0                         0
PAGE TMS SEGHDR          5                        19
4                   0                         73.6842105263158
PAGE TMS LFBMP           0                         0
0                   0                         0
PAGE TMS ITBMP           0                         0
0                   0                         0
PAGE TMS RTBMP           0                         0
0                   0                         0
PAGE TMS EXTDIR          0                         0
0                   0                         0
PAGE CMS SEGHDR          0                      1536
0                 512                       100
PAGE CMS EXTDIR          0                         0
0                   0                         0
PAGE FEBT FSB            2                      1024
515                   2                        99.8046875
PAGE FEBT EGH            0                       512
0                   4                       100
PAGE LOB META            0                         0
0                   0                         0
PAGE HV TEMP NODE        0                         0
0                   0                         0
26 rows selected.
```

V$BUFFPOOL_STAT

This view displays statistics including the buffer pool hit ratio and the buffer control block (BCB) count of the buffer pool.

Column name	Type	Description
ID	INTEGER	The identifier of the buffer pool
POOL_SIZE	INTEGER	The number of pages in the buffer pool
PAGE_SIZE	INTEGER	The size of a page (in bytes)
HASH_BUCKET_COUNT	INTEGER	The number of hash table buckets
HASH_CHAIN_LATCH_COUNT	INTEGER	The number of chain latches used in the hash table of the buffer pool
LRU_LIST_COUNT	INTEGER	The number of LRU lists

Column name	Type	Description
PREPARE_LIST_COUNT	INTEGER	The number of prepare lists in the buffer pool
FLUSH_LIST_COUNT	INTEGER	The number of flush lists in the buffer pool
CHECKPOINT_LIST_COUNT	INTEGER	The number of checkpoint lists in the buffer pool
VICTIM_SEARCH_COUNT	INTEGER	The number of victim searches in an LRU List
HASH_PAGES	INTEGER	The number of pages inserted into the hash table at present
HOT_LIST_PAGES	INTEGER	The number of pages in LRU hot lists at present
COLD_LIST_PAGES	INTEGER	The number of pages in LRU cold lists at present
PREPARE_LIST_PAGES	INTEGER	The number of prepare lists in the buffer pool
FLUSH_LIST_PAGES	INTEGER	The number of pages in all flush lists at present
CHECKPOINT_LIST_PAGES	INTEGER	The number of pages in all checkpoint lists at present
FIX_PAGES	BIGINT	The accumulated number of page fix requests without latches
GET_PAGES	BIGINT	The accumulated number of page requests for which latches were obtained
READ_PAGES	BIGINT	The accumulated number of page reads from disk
CREATE_PAGES	BIGINT	The accumulated number of new page creation tasks
HIT_RATIO	DOUBLE	The cumulative hit ratio from the buffer pool since the system was started
HOT_HITS	BIGINT	The accumulated number of accesses to an LRU hot list
COLD_HITS	BIGINT	The accumulated number of accesses to an LRU cold list
PREPARE_HITS	BIGINT	The accumulated number of accesses to a prepare list
FLUSH_HITS	BIGINT	The accumulated number of accesses to a prepare list
OTHER_HITS	BIGINT	The accumulated number of accesses to buffers not included on any list

Column name	Type	Description
PREPARE_VICTIMS	BIGINT	The accumulated number of searches for replacement targets on a prepare list
LRU_VICTIMS	BIGINT	The accumulated number of searches for replacement targets on an LRU list
VICTIM_FAILS	BIGINT	The number of failures to find a replacement target
PREPARE_AGAIN_VICTIMS	BIGINT	The cumulative number of searches for a replacement target buffer on a prepare list after failing to find a replacement target on an LRU list
VICTIM_SEARCH_WARP	BIGINT	The number of searches that continued to subsequent prepare lists after failing to find replacement targets on prepare lists or LRU lists
LRU_SEARCHS	BIGINT	The accumulated number of searched buffers on an LRU list
LRU_SEARCHS_AVG	INTEGER	The average number of buffers searched for a replacement target
LRU_TO_HOTS	BIGINT	The accumulated number of times that a Buffer Control Block (BCB) has moved into a hot area in an LRU list
LRU_TO_COLDS	BIGINT	The accumulated number of times that a BCB has moved into a cold area in an LRU list
LRU_TO_FLUSHS	BIGINT	The accumulated number of times that a BCB has moved from an LRU list to a flush list
HOT_INSERTIONS	BIGINT	The accumulated number of insertions into LRU hot lists
COLD_INSERTIONS	BIGINT	The accumulated number of insertions into LRU cold lists
DB_SINGLE_READ_PERF	DOUBLE	The average number of bytes that are read from disk per second when one data page is read from a disk data file
DB_MULTI_READ_PERF	DOUBLE	The average number of bytes that are read per second when multiple data pages are read from a disk data file at the same time

Column Information

ID

This is a unique buffer pool number. Its value is 0 because multiple buffer pools are not currently supported.

POOL_SIZE

This is the number of pages in the buffer pool. POOL_SIZE * PAGE_SIZE is equal to the size specified by the BUFFER_AREA_SIZE property.

PAGE_SIZE

This is the size of the pages used in the buffer pool at present. Only the fixed value 8192 is possible, because multiple buffer pools are not currently supported.

HASH_BUCKET_COUNT

This is the number of hash table buckets. It is determined by the BUFFER_HASH_BUCKET_DENSITY property. This value cannot be changed while the server is running. The greater this value is, the less expensive it is to search the hash bucket list.

HASH_CHAIN_LATCH_COUNT

This is the number of chain latches used in the hash table. The greater this value is, the less competition there is for latches, which can occur when searching the hash table.

LRU_LIST_COUNT

This is the number of LRU lists in the buffer pool.

PREPARE_LIST_COUNT

This is the number of prepare lists in the buffer pool.

FLUSH_LIST_COUNT

This is the number of flush lists in the buffer pool.

CHECKPOINT_LIST_COUNT

This is the number of flush lists in the buffer pool.

VICTIM_SEARCH_COUNT

This is the maximum number of BCBs that are searched when searching for replacement targets in LRU lists. If the search for replacement targets reaches the specified value and no replacement target is found, Buffer Manager waits until the flusher adds a clean buffer to the prepare list.

HASH_PAGES

This is the number of buffers that have been inserted into the hash table. Its value indicates the number of buffers currently in use.

HOT_LIST_PAGES

This is the number of buffers that exist on the LRU hot list.

COLD_LIST_PAGES

This is the number of buffers that exist on the LRU cold list.

PREPARE_LIST_PAGES

This is the number of buffers that exist on the prepare list. If the value is 0, the LRU list is searched in order to obtain replacement targets.

FLUSH_LIST_PAGES

This is the number of buffers that exist on the flush list. A high value means that there are many buffers to be flushed.

CHECKPOINT_LIST_PAGES

This is the number of buffers that exist on the checkpoint list. It also indicates the number of pages that have been renewed.

FIX_PAGES

This is the cumulative number of pages that have been requested without obtaining latches since the system was started.

GET_PAGES

This is the cumulative number of page latches that have been have been requested and obtained since the system was started.

READ_PAGES

This is the cumulative number of pages that have been read from disk when requesting a page. It also indicates the number of buffer misses.

CREATE_PAGES

This is the cumulative number of page assignments for the insertion of data into new pages. Page creation isn't actually accompanied by disk I/O.

HIT_RATIO

This is the cumulative hit ratio in the buffer pool. It can be calculated thus: (GET_PAGES + FIX_PAGES - READ_PAGES)/(GET_PAGES + FIX_PAGES). If this value is low, it means that many pages have been read from disk instead of from the cache. In other words, if the value is low, the system will not be able to process queries quickly.

HOT_HITS

This is the cumulative number of hits on the LRU hot list. If a requested page is already in the buffer, a hit doesn't cause a page to be read.

COLD_HITS

This is the cumulative number of hits on the LRU cold list.

PREPARE_HITS

This is the cumulative number of hits on the prepare list.

FLUSH_HITS

This is the cumulative number of hits on the flush list.

OTHER_HITS

This is the number of hits on a buffer that was not on any list at that moment. A hit buffer need not always be on a list.

PREPARE_VICTIMS

This is the cumulative number of searches for replacement buffers on a prepare list.

LRU_VICTIMS

This is the cumulative number of searches for replacement buffers on an LRU list.

VICTIM_FAILS

This is the cumulative number of failures to find a replacement target buffer. This value can be calculated thus: PREPARE_AGAIN_VICTIMS + VICTIM_SEARCH_WARP.

Summing PREPARE_VICTIMS + LRU_VICTIMS + VICTIM_FAILS gives the total number of replacements in the buffer pool.

PREPARE_AGAIN_VICTIMS

After failing to find replacement target buffers, it is necessary to wait for the insertion of buffers on a prepare list. While waiting, this is the number of clean buffers that have been received and selected as replacement targets.

VICTIM_SEARCH_WARP

This is the cumulative number of searches for replacement target buffers that failed after the specified period of time and thus passed to the next prepare list.

LRU_SEARCHS

This is the cumulative number of buffers for which searches for replacement target buffers have been made in the LRU list.

LRU_SEARCHS_AVG

This is the average number of buffers that are searched when searching for a replacement target.

LRU_TO_HOTS

This is the cumulative number of times that buffers have moved into hot areas in LRU lists.

LRU_TO_COLDS

This is the cumulative number of times that buffers have moved into cold areas in LRU lists.

LRU_TO_FLUSHS

This is the cumulative number of times that buffers have moved from LRU lists to flush lists.

HOT_INSERTIONS

This is the cumulative number of insertions into LRU hot lists.

COLD_INSERTIONS

This is the cumulative number of insertions into LRU cold lists.

DB_SINGLE_READ_PERF

When FETCH, INSERT, UPDATE and DELETE operations are performed on disk tables, one data page is read from a data file on disk and stored in a memory buffer. This is the average number of bytes that are read from disk per second (in kB/sec) in the course of such tasks.

DB_MULTI_READ_PERF

When a so-called "full scan" is performed, i.e. when an entire disk table is scanned, multiple data pages are simultaneously read from a data file on disk and stored in a memory buffer. This is the average number of bytes that are read from disk per second (in kB/sec) in the course of this task.

V$CATALOG

This view displays information about the structure of the tables that exist in the database.

Column name	Type	Description
TABLE_OID	BIGINT	The object identifier of the table
COLUMN_CNT	INTEGER	The number of columns in the table
COLUMN_VAR_SLOT_CNT	INTEGER	The number of variable slots, which are used to store information about columns
INDEX_CNT	INTEGER	The number of indexes in the table
INDEX_VAR_SLOT_CNT	INTEGER	The number of variable slots, which are used to store information about indexes

Column Information

TABLE_OID

This is the physical location of the header, which contains information about the table.

COLUMN_CNT

This is the number of columns in the table.

COLUMN_VAR_SLOT_CNT

This is the number of variable slots, which are used to store information about the columns in the table.

INDEX_CNT

This is the number of indexes in the table.

INDEX_VAR_SLOT_CNT

This is the number of variable slots, which are used to store information about the indexes in the table.

V$DATABASE

V$DATABASE displays internal information about the memory database.

Column name	Type	Description
DB_NAME	VARCHAR(128)	The database name
PRODUCT_SIGNATURE	VARCHAR(512)	A string describing the product binary and build environment
DB_SIGNATURE	VARCHAR(512)	A unique database identification string
VERSION_ID	INTEGER	The version of the database
COMPILE_BIT	INTEGER	Whether the product was compiled for 32 bits or 64 bits
ENDIAN	BIGINT	Endian information
LOGFILE_SIZE	BIGINT	The log file size
TX_TBL_SIZE	INTEGER	The transaction table size
LAST_SYSTEM_SCN	VARCHAR(29)	For internal usage only
INIT_SYSTEM_SCN	VARCHAR(29)	For internal usage only
DURABLE_SYSTEM_SCN	VARCHAR(29)	The saved system SCN value
MEM_MAX_DB_SIZE	VARCHAR(256)	The maximum size of the memory database

Column name	Type	Description
MEM_ALLOC_PAGE_COUNT	BIGINT	The total number of allocated pages
MEM_FREE_PAGE_COUNT	BIGINT	The total number of available pages
MAX_ACCESS_FILE_SIZ	VARCHAR(12)	The maximum file size that can be created in the database

Column Information

DB_NAME

This is the name of the memory database.

PRODUCT_SIGNATURE

This is unique product information about Altibase.

DB_SIGNATURE

A unique database identification string.

VERSION_ID

This is a unique version number managed by the storage manager of Altibase.

COMPILE_BIT

This indicates whether the database was compiled as a 32-bit or 64-bit application.

ENDIAN

This is the Endian of the database.

- 0: little endian
- 1: big endian

LOGFILE_SIZE

This is the size, in bytes, of the log files used by the database.

TX_TBL_SIZE

This is the size of the transaction table.

MEM_MAX_DB_SIZE

This is the maximum size to which the memory database can expand.

MEM_ALLOC_PAGE_COUNT

This is the total number of pages currently allocated to the memory database. This only indicates the current size of memory database space, not the maximum size to which it can expand. The current size of memory database space can be calculated by multiplying the sum of MEM_ALLOC_PAGE_COUNT and MEM_FREE_PAGE_COUNT by the page size (32kB).

MEM_FREE_PAGE_COUNT

This is the number of pages available to be allocated to memory database space, not including the number of pages that are currently allocated. This only pertains to the current size of memory database space, not the maximum size to which it can expand. The current size of memory database space can be calculated by multiplying the sum of MEM_ALLOC_PAGE_COUNT and MEM_FREE_PAGE_COUNT by the page size (32kB).

DURABLE_SYSTEM_SCN

This is the system SCN value saved in database.

V$DATAFILES

This view displays information about the data files used in tablespaces.

Column name	Type	Description
ID	INTEGER	The data file identifier
NAME	VARCHAR(256)	Data file name
SPACEID	INTEGER	The tablespace identifier
OLDEST_LSN_LFGID	INTEGER	Not used (0)
OLDEST_LSN_FILENO	INTEGER	See below
OLDEST_LSN_OFFSET	INTEGER	See below
CREATE_LSN_LFGID	INTEGER	Not used (0)
CREATE_LSN_FILENO	INTEGER	See below
CREATE_LSN_OFFSET	INTEGER	See below
SM_VERSION	INTEGER	Version information
NEXTSIZE	BIGINT	The size at the next increase
MAXSIZE	BIGINT	The maximum size
INITSIZE	BIGINT	The initial size
CURRSIZE	BIGINT	The current size
AUTOEXTEND	INTEGER	An auto-extension flag
IOCOUNT	INTEGER	The number of I/O operations currently underway
OPENED	INTEGER	Indicates whether or not the file is currently in use
MODIFIED	INTEGER	Indicates whether or not the file is currently being modified

Column name	Type	Description
STATE	INTEGER	The status of the file
MAX_OPEN_FD_COUNT	INTEGER	The maximum number of FDs that can be opened
CUR_OPEN_FD_COUNT	INTEGER	The number of open FDs

Column Information

ID

This is the identifier of the data file. In order to avoid duplicate identifiers, identifiers are assigned sequentially in the order in which data files are created.

NAME

This is the physical path and name of the data file.

SPACEID

This is the identifier of the tablespace containing the data file.

OLDEST_LSN_FILENO

This is the file number portion of the LSN value of the oldest of the pages that were loaded into the buffer and changed at the time of the last checkpoint, when pages in the data file were flushed to disk.

OLDEST_LSN_OFFSET

This is the offset value portion of the LSN value of the oldest of the pages that were loaded into the buffer and changed at the time of the last checkpoint, when pages in the data file were flushed to disk.

CREATE_LSN_FILENO

This is the file number portion of the LSN that was current at the time at which the data file was created.

CREATE_LSN_OFFSET

This is the offset value portion of the LSN that was current at the time at which the data file was created.

SM_VERSION

This is the version of the binary from which the data file was created.

NEXTSIZE

If the data file's autoextend property is set to "on", this is the size by which the data file will be increased when there is insufficient space. (1 page = 8kB)

MAXSIZE

If the data file's autoextend property is set to "on", this is the maximum size to which the data file can be increased when there is insufficient space. (1 page = 8kB)

INITSIZE

This is the initial size of the data file at the time of its creation
(1 page = 8kB).

CURRSIZE

This is the current size of the data file. (1 page = 8kB).

AUTOEXTEND

This indicates whether the size of the data file will be increased automatically when there is insufficient space.

- 0: No automatic increase
- 1: Automatic increase

IOCOUNT

This is the number of I/O operations currently underway on the data file. If no data I/O is in progress on the data file, the next data file can be opened.

OPENED

This indicates whether the data file is currently open.

- 0: Closed
- 1: Opened

MODIFIED

This indicates whether the data file has been modified. If any pages have been flushed to the data file without subsequent synchronization, this value is 1. if synchronization has been executed on the data file since pages were last flushed to it, this value is 0.

STATE

This is the status of the data file.

- 1: Offline
- 2: Online
- 6: Backup is in progress
- 128: Dropped

MAX_OPEN_FD_COUNT

This is the maximum number of FDs (File Descriptors) that can be opened when performing I/O on the current disk data file.

CUR_OPEN_FD_COUNT

This is the number of open FDs (File Descriptors) for the current disk data file.

V$DATATYPE

This table shows information about the data types that are supported by Altibase.[14]

[14] The value stored in this performance view is the value retrieved by the ODBCSQLQGettypeInfo () function.

For more detailed information, please refer to the *ODBC Reference*.

Column name	Type	Description
TYPE_NAME	VARCHAR(40)	The name of a data type that is supported in the DBMS
DATA_TYPE	SMALLINT	An internally defined value indicating a data type that is supported in the DBMS
ODBC_DATA_TYPE	SMALLINT	The identifier of an ODBC SQL data type corresponding to the data type
COLUMN_SIZE	INTEGER	The maximum column size for the data type
LITERAL_PREFIX	VARCHAR(4)	Characters recognized as the prefix of the data type literal
LITERAL_SUFFIX	VARCHAR(4)	Characters recognized as the suffix of the data type literal
CREATE_PARAM	VARCHAR(20)	When using SQL to define a data type, a parameter keyword list enclosed in parentheses
NULLABLE	SMALLINT	Indicates whether NULL values are allowed for the data type
CASE_SENSITIVE	SMALLINT	Indicates whether the data type is case-sensitive
SEARCHABLE	SMALLINT	Indicates how the data type is used in a WHERE clause
UNSIGNED_ATTRIBUTE	SMALLINT	For a numeric data type, indicates whether the data type is a signed data type
FIXED_PREC_SCALE	SMALLINT	Indicates whether the data type is a fixed type

Column name	Type	Description
AUTO_UNIQUE_VALUE	SMALLINT	Reserved for future use
LOCAL_TYPE_NAME	VARCHAR(40)	The name of the data type in the local language
MINIMUM_SCALE	SMALLINT	The minimum allowable number of digits to the right of the decimal point
MAXIMUM_SCALE	SMALLINT	The maximum allowable number of digits to the right of the decimal point
SQL_DATA_TYPE	SMALLINT	A defined value of a SQL data type that is provided by SQL_DESC_TYPE in ODBC
SQL_DATETIME_SUB	SMALLINT	A type subcode for a datetime or interval data type
NUM_PREC_RADIX	INTEGER	The number of bits that are needed to perform operations on the maximum number of digits that a column can hold
INTERVAL_PRECISION	SMALLINT	When the DATA_TYPE is interval, the maximum number of digits needed to express the data

Column Information

ODBC_DATA_TYPE

This is the data type identifier for the ODBC SQL data type corresponding to the data type. For more information, please refer to the appendix pertaining to data types in the *ODBC Reference*.

COLUMN_SIZE

This is the maximum column size for the data type.

For numeric data types, this is the precision value, which was specified when the type was defined. For string data types, this is the length value, which was specified when the type was defined. For datetime data types, this is the total number of characters that are needed to display a value when it is converted to characters.

LITERAL_PREFIX

This specifies the characters that signify the prefix of a literal for the data type. For data types to which literal prefixes do not apply, it is NULL.

LITERAL_SUFFIX

This specifies the characters that signify the suffix of a literal for the data type. For data types to which literal suffixes do not apply, it is NULL.

CREATE_PARAM

When using SQL to define a data type, this is a comma-separated list of parameter keywords enclosed in parentheses. For example, to express a NUMBER as NUMBER(precision,scale), the content within the parentheses, that is, "precision, scale", is the list. "Precision" and "scale" are thus both keywords in the list. For data types that do not need parameters, this is set to NULL.

NULLABLE

This indicates whether NULL values are allowed for a data type.

- 1: NULL is allowed.
- 0: NULL is not allowed.

CASE_SENSITIVE

For character data types, indicates whether to distinguish between uppercase and lowercase letters when sorting data of the data type.

- 1: Case-sensitive.
- 0: Not case-sensitive.

SEARCHABLE

Indicates how a data type can be used in a WHERE clause.

- 0: It cannot be used in a WHERE clause (SQL_PRED_NONE).
- 1: It can be used in a WHERE clause, but must be used with LIKE (SQL_PRED_CHAR).
- 2: It can be used in a WHERE clause with any comparison operator except LIKE (SQL_PRED_BASIC).
- 3: It can be used in a WHERE clause with any comparison operator (SQL_SEARCHABLE).

UNSIGNED_ATTRIBUTE

Indicates whether a data type is signed.

- 1: The data type is an unsigned data type.
- 0: The data type is a signed data type.
- NULL: The data type is not numeric, therefore this attribute is not applicable.

FIXED_PREC_SCALE

Indicates whether a data type is fixed. If a data type is a fixed numeric type and always has the same precision and scale, this value is 1 (SQL_TRUE). Otherwise, it is 0 (SQL_FALSE).

LOCAL_TYPE_NAME

Indicates a localized (region-specific) name for a data type. If there is no localized name, this value is NULL.

MINIMUM_SCALE

For numeric data types, this is the minimum allowable number of digits to the right of the decimal. This value exists for fixed scale types; it is set to NULL for types to which scale does not pertain

MAXIMUM_SCALE

For numeric data types, this is the maximum allowable number of digits to the right of the decimal. It is specified when the data type is defined. It is set to NULL for types to which scale does not pertain.

SQL_DATA_TYPE

This is a SQL data type that is provided by SQL_DESC_TYPE in ODBC. For data types other than INTERVAL or DATETIME, this value is the same as that of ODBC_DATA_TYPE.

SQL_DATETIME_SUB

If the SQL_DATA_TYPE value is SQL_DATETIME or SQL_INTERVAL, this is the type sub code for the DATETIME or INTERVAL data type. If the data type is not DATETIME or INTERVAL, it is set to NULL.

NUM_PREC_RADIX

This is the number of bits or digits that are needed to perform mathematical operations on the highest number that a column can hold

INTERVAL_PRECISION

This is the maximum number of digits that a DATA_TYPE of type INTERVAL can hold.

V$DBA_2PC_PENDING

This view shows a list of XIDs (transaction IDs) for distributed transactions that exist in the DBMS and whose status is in doubt. The status of a distributed transaction is said to be "in-doubt" when a branch thereof is ready to be committed, but has not yet been committed or rolled back.

Column name	Type	Description
LOCAL_TRAN_ID	BIGINT	An internal Altibase transaction identifier that is associated with the GLOBAL_TX_ID
GLOBAL_TX_ID	VARCHAR(256)	Globally unique transaction identifier

Column Information

LOCAL_TRAN_ID

This is an internal Altibase transaction identifier that is associated with a global transaction identifier.

GLOBAL_TX_ID

This is the globally unique transaction identifier. The GLOBAL_TX_ID contains a format identifier, two length fields and a data field. The data field consists of at most two contiguous components: a global transaction identifier and a branch qualifier.

V$DBLINK_ALTILINKER_STATUS

This view shows status information about the AltiLinker process for the database link.

Column name	Type	Description
STATUS	INTEGER	Status of the AltiLinker process. Refer to [Column Information]().
SESSION_COUNT	INTEGER	The number of linker sessions, the sessions between Altibase and the Altilinker process.
REMOTE_SESSION_COUNT	INTEGER	The number of sessions between the Altilinker process and the remote servers
JVM_MEMORY_POOL_MAX_SIZE	INTEGER	The maximum size of the memory pool allocated for the AltiLinker on the JVM
JVM_MEMORY_USAGE	BIGINT	The amount of memory used for the AltiLinker process on the JVM
START_TIME	VARCHAR(128)	The date and time at which the Altilinker process started

Column Information

STATUS

This is the status of the Altilinker.

- 0 : The AltiLinker process has not started or is in an abnormal state.
- 1 : The AltiLinker process is started.
- 2 : A Linker Control Session is created between the AltiLinker process and the Altibase server, and AltiLinker is running normally.

V$DBLINK_DATABASE_LINK_INFO

This view displays information about the database link object existing in the database.

Column name	Type	Description
ID	INTEGER	The database link object identifier
STATUS	INTEGER	The status of the database link object
REFERENCE_COUNT	INTEGER	The number of references of the database link object

Column Information

STATUS

Displays the status of the database link object.

- 1(CREATED): Creation of the database link object is complete.
- 2(META): Registration of the database link object information in the meta table.
- 3(READY): The database link object is ready for use.

REFERENCE_COUNT

Displays the number of times the database link is currently being referenced.

V$DBLINK_GLOBAL_TRANSACTION_INFO

This view displays information about global transactions being executed through the current database link.

Column name	Type	Description
TRANSACTION_ID	INTEGER	The identifier of the global transaction that is currently using the database link
STATUS	INTEGER	The current status of the global transaction
SESSION_ID	INTEGER	The ID of the Linker Data Session executing the global transaction
REMOTE_TRANSACTION_COUNT	INTEGER	The number of remote transactions currently being executed within the global transaction
TRANSACTION_LEVEL	INTEGER	The execution level of the global transaction
GLOBAL_TRANSACTION_ID	INTEGER	The global transaction identifier using the database link

Column Information

STATUS

Displays the current state of the global transaction.

- 0(NONE): No transaction exists.
- 1(BEGIN): The global transaction has started.
- 2(PREPARE_READY): The global tranaction has started, however, no remote transaction under execution exists.
- 3(PREPARE_REQUEST): The AltiLinker process has been requested to PREPARE at the Simple Transaction Commit level.
- 4(PREPARE_WAIT): The global transaction is waiting for all remote transactions to complete PREPARE at the Simple Transaction Commit level.

- 5(PREPARED): All remote transactions have completed PREPARE.

- 6(COMMIT_REQUEST): COMMIT has been requested via the AltiLinker process

- 7(COMMIT_WAIT): Waiting for a response on COMMIT from the AltiLinker process.

- 8(COMMITTED): The global transaction is committed

- 9(ROLLBACK_REQUEST): ROLLBACK has been requested to the AltiLinker process.

- 10(ROLLBACK_WAIT): AWaiting for a response on ROLLBACK from the AltiLinker process.

- 11(ROLLBACKED): The global transaction is rolled back.

TRANSACTION_LEVEL

Displayed as 0, 1, or 2. For further information about each value, please refer to the DBLINK_GLOBAL_TRANSACTION_LEVEL property.

V$DBLINK_LINKER_CONTROL_SESSION_INFO

This view displays status information about the Linker Control Session which is singularly created for the control operations between the Altibase server and the AltiLinker process.

Column name	Type	Description
STATUS	INTEGER	The status of the Linker Control Session
REFERENCE_COUNT	INTEGER	The number of times the Linker Control Session is currently being referenced

Column Information

STATUS

Displays the current status of the Linker Control Session.

- 0(NONE): No Linker Control Session exists.

- 1(CREATED): Creation of a Linker Control Session is complete.

- 2(CONNECTED): The AltiLinker process and the Linker Control Session are connected.

- 3(DISCONNECTED): The AltiLinker process and the Linker Control Session are disconnected.

- 5(LOCKED): The Linker Control Session is locked.

- 6(UNLOCKED): The Linker Control Session is unlocked.

V$DBLINK_LINKER_DATA_SESSION_INFO

This view displays the status information about Linker Data Sessions created for the execution of data operations between the Altibase server and the AltiLinker process

Column name	Type	Description
ID	INTEGER	The Linker Data Session identifier

Column name	Type	Description
STATUS	INTEGER	The status of the Linker Data Session
LOCAL_TRANSACTION_ID	INTEGER	The local transaction identifier executing in the current session
GLOBAL_TRANSACTION_ID	INTEGER	The global transaction identifier executing in the current session

Column Information

STATUS

Displays the current status of the Linker Data Session.

- 0(NONE): No Linker Data Session exists.
- 1(CREATED): Creation of the Linker Data Session is complete.
- 2(CONNECTED): The Linker Data Session and the AltiLinker process are connected.
- 3(DISCONNECTED): The Linker Data Session and the AltiLinker process are disconnected.
- 4(DESTROYED): The Linker Data Session has been removed.
- 5(LOCKED): The Linker Control Session has been locked
- 6(UNLOCKED): The Linker Control Session is unlocked.

V$DBLINK_LINKER_SESSION_INFO

This view displays how many Linker Control Sessions and Linker Data Sessions exist between the Altibase server and the AltiLinker process.

Column name	Type	Description
SESSION_ID	INTEGER	The linker session identifier
STATUS	INTEGER	The status of the linker session
SESSION_TYPE	VARCHAR(7)	Indicates whether it is a Linker Control Session or a Linker Data Session

Column Information

STATUS

Indicates the current status of the Linker Session. For the status value, please refer to the STATUS of the performance views, V$DBLINK_LINKER_CONTROL_SESSION_INFO and V$DBLINK_LINKER_DATA_SESSION_INFO.

SESSION_TYPE

Indicates whether the linker session is a Linker Control Session or a Linker Data Session.

- CONTROL: Linker Control Session
- DATA: Linker Data Session

V$DBLINK_NOTIFIER_TRANSACTION_INFO

This view displays information on hte distributed transaction AltiLinker is processing.

Column name	Type	Description
GLOBAL_TRANSACTION_ID	INTEGER	The transaction identifier using the database link.
TRANSACTION_ID	INTEGER	The local transaction identifier
XID	VARCHAR(12)	The transaction branch identifier
TRANSACTION_RESULT	VARCHAR(10)	The result of transaction process(COMMIT/ROLLBACK).
TARGET_INFO	VARCHAR(40)	The name of remote server that an object of the database link will be accessing.

Column Information

GLOBAL_TRANSACTION_ID

This is the identifier of the global transaction which uses the database link.

TRANSACTION_ID

This is the inner transaction identifier when Altibase performs the local transaction in case of processing the global transaction.

XID

This is the transaction ID allocated to the transaction branch. The value displays the format identifier, global transaction identifier, or branch qualifier

TRANSACTION_RESULT

This information indicates the result of a processed transaction.

- COMMIT: The transaction is processed by COMMIT.
- ROLLBACK: P The transaction is processed by ROLLBACK.TARGET_INFO

This shows the name of the remote server that the database link object will access.

V$DBLINK_REMOTE_STATEMENT_INFO

This view displays information about information of the query occurred in the remote server when using the database link.

Column name	Type	Description
TRANSACTION_ID	INTEGER	The transaction identifier of using the database link.
REMOTE_TRANSACTION_ID	INTEGER	The transaction identifier occurred in the remote server.
STATEMENT_ID	BIGINT	The statement identifier occurred in the remote server.
QUERY	VARCHAR(32000)	The query contents executed in the statement.
GLOBAL_TRANSACTION_ID	INTEGER	The global transaction identifier which is using the database link.

Column Information

REMOTE_TRANSACTION_ID

This is the identifier of the transaction which occurred on the remote server. This identifier is not the transaction identifier which was actually created on the remote server, but is an identifier autonomously granted by AltiLinker while executing a transaction through a remote server. Since this identifier is created for managemenet purposes, the value is of little significance.

STATEMENT_ID

This is the statement identifier that occurred on the remote server. This identifier is not a statement identifier actually generated at the remote server, but an identifier that AltiLinker assigns itself when generating a sentence at the remote server.

V$DBLINK_REMOTE_TRANSACTION_INFO

This view displays information about all remote transactions being executed on the remote node through the database link.

Column name	Type	Description
TRANSACTION_ID	INTEGER	The identifier of the transaction which is using the database link
REMOTE_TRANSACTION_ID	INTEGER	The identifier of the transaction which occurred on the remote server
TARGET_INFO	VARCHAR(40)	The remote server name

Column name	Type	Description
STATUS	INTEGER	The current status of the global transaction
XID	VARCHAR(12)	The identifier of the transaction branch
GLOBAL_TRANSACTION_ID	INTEGER	The global transaction identifier which is using the database link.

Column Information

REMOTE_TRANSACTION_ID

This is the identifier of the transaction which occurred on the remote server. This identifier is not the transaction identifier which was actually created on the remote server, but is an identifier autonomously granted by AltiLinker while executing a transaction through a remote server. Since this identifier is created for management purposes, the value is of little significance.

STATUS

Displays the current status of the global transaction.

- 0(NONE): No transaction exists
- 1(BEGIN): A transaction has started.
- 2(PREPARE_READY): A transaction has started, however, no remote transaction under execution exists.
- 3(PREPARE_WAIT): The global transaction is waiting for a response on PREPARE from the AltiLinker process at the Simple Transaction Commit Level.
- 4(PREPARED): PREPARE is complete.
- 5(COMMIT_WAIT): Waiting for a response on COMMIT from the AltiLinker process.
- 6(COMMITTED): The global transaction is committed.
- 7(ROLLBACK_WAIT): Waiting for a response on ROLLBACK from the AltiLinker process.
- 8(ROLLBACKED): The global transaction is rolled back.

V$DBMS_STATS

This view displays statistical information about the whole database.

Column name	Type	Description
DATE	CHAR(48)	The time at which statistical information was collected for the last time
SAMPLE_SIZE	DOUBLE	The sample size
NUM_ROW_CHANGE	BIGINT	The change in the number of rows after statistical information was collected for the last time

Column name	Type	Description
TYPE	CHAR(1)	The type of the statistics target: S: System T: Table I: Index C: Column
SREAD_TIME	DOUBLE	The amount of time spent on reading one page
MREAD_TIME	DOUBLE	The amount of time spent on reading multiple pages at a time
MREAD_PAGE_COUNT	BIGINT	The number of pages read when multiple pages are read at a time
HASH_TIME	DOUBLE	The average execution time for hashing
COMPARE_TIME	DOUBLE	The average execution time for comparing
STORE_TIME	DOUBLE	The average execution time for storing memory temporary tables
TARGET_ID	BIGINT	The OID of the statistics target table or the ID of the statistics target index
COLUMN_ID	INTEGER	The ID of the statistics target column
NUM_ROW	BIGINT	The number of rows
NUM_PAGE	BIGINT	The number of pages
NUM_DIST	BIGINT	The number of distinct rows
NUM_NULL	BIGINT	The number of NULLs
AVG_LEN	BIGINT	The average length of rows or column data
ONE_ROW_READ_TIME	DOUBLE	The average time spent on reading one row
AVG_SLOT_COUNT	BIGINT	The average number of slots per leaf node
INDEX_HEIGHT	BIGINT	The depth from the root node to the leaf node in the index
CLUSTERING_FACTOR	BIGINT	The degree to which data is sorted according to the index
MIN	CHAR(48)	The minimum value
MAX	CHAR(48)	The maximum value
META_SPACE	BIGINT	The amount of space used for data management
USED_SPACE	BIGINT	The amount of space used for data storage

Column name	Type	Description
AGEABLE_SPACE	BIGINT	The amount of space available for reuse due to aging in the future
FREE_SPACE	BIGINT	The amount of space available for use

Column Information

DATE

This is the time at which statistical information was collected for the last time.

SAMPLE_SIZE

This is the sample size chosen for the collection of statistical information.

NUM_ROW_CHANGE

This is the change in the number of rows after statistical information was collected for the last time.

TYPE

This is the type of statistics target for collection, and takes one of the following values:

S: System

T: Table

I: Index

C: Column

SREAD_TIME

This is the average time spent on reading one page.

MREAD_TIME

This is the average time spent on reading multiple pages at a time.

MREAD_PAGE_COUNT

This is the number of pages specified to be read for reading multiple pages at a time.

HASH_TIME

This is the average execution time spent on hashing.

COMPARE_TIME

This is the average execution time spent on comparing.

STORE_TIME

This is the average time spent on storing to a memory temporary table.

TARGET_ID

This is the OID of the statistics target table for collection or the ID of the statistics target index for collection.

COLUMN_ID

This is the statistics target column ID for collection.

NUM_ROW

This is the number of rows of the statistics target for collection (a tables or an index).

NUM_PAGE

This is the number of pages of the statistics target for collection (a table or an index).

NUM_DIST

This is the number of distinct values of the index or the column.

NUM_NULL

This is the number of NULL values of the column.

AVG_LEN

This is the average length of row data, if the statistics target for collection is a table. This is the average length of column data, if the statistics target for collection is a column.

ONE_ROW_READ_TIME

This is the average time spent on reading one row.

AVG_SLOT_COUNT

This is the average number of slots per leaf node.

INDEX_HEIGHT

This is the depth from the root node to the lead node in the index.

CLUSTERING_FACTOR

This is the degree to which data is sorted according to the index.

MIN

This is the minimum value of the index or the column.

MAX

This is the maximum value of the index or the column.

META_SPACE

This is the amount of space used for data management.

USED_SPACE

This is the amount of space used for data storage.

AGEABLE_SPACE

This is the amount of space available for reuse due to aging in the future.

FREE_SPACE

This is the amount of space available for use among the space allocated to the table or the index.

V$DB_FREEPAGELISTS

This view displays information about lists of pages that can be used, that is, free pages, in a database.

Column name	Type	Description
SPACE_ID	INTEGER	The identifier of the tablespace to which the free pages belong
RESOURCE_GROUP_ID	INTEGER	The identifier of the resource group
FIRST_FREE_PAGE_ID	INTEGER	The identifier of the first free page in the list
FREE_PAGE_COUNT	BIGINT	The total number of free pages in the list

Column Information

RESOURCE_GROUP_ID

This is a unique number that is used to identify the list.

FIRST_FREE_PAGE_ID

This is the identifier of the first free page in the list.

FREE_PAGE_COUNT

This is the number of free pages on the list.

V$DB_PROTOCOL

This view shows information about Altibase communication protocols of all incoming packets.

Column name	Type	Description

Column name	Type	Description
OP_NAME	VARCHAR(50)	The protocol name
OP_ID	INTEGER	The unique identifier of the protocol
COUNT	BIGINT	The cumulative number of incoming packets for this protocol

V$DIRECT_PATH_INSERT

This view displays historical statistics on direct-path uploads.

Column name	Type	Description
COMMIT_TX_COUNT	BIGINT	The total number of transactions that were successfully committed using the direct-path option
ABORT_TX_COUNT	BIGINT	The total number of transactions that were rolled back while data were being uploaded using the direct-path option
INSERT_ROW_COUNT	BIGINT	The total number of rows that were inserted by iLoader using the direct-path option
ALLOC_BUFFER_PAGE_TRY_COUNT	BIGINT	The total number of times that page allocation was requested
ALLOC_BUFFER_PAGE_FAIL_COUNT	BIGINT	The total number of times that a page allocation request failed

Column Information

COMMIT_TX_COUNT

This is the total number of transactions which were committed by iLoader using the direct-path option, accumulated over past executions.

ABORT_TX_COUNT

This is the total number of transactions which were rolled back due to errors while data were being uploaded using the direct-path option, accumulated over past executions.

INSERT_ROW_COUNT

This is the total number of rows which were inserted by iLoader using the direct-path option, accumulated over past executions.

ALLOC_BUFFER_PAGE_TRY_COUNT

This is the total number of times that page allocation was requested for uploading data using the direct-path option, accumulated over past executions.

ALLOC_BUFFER_PAGE_FAIL_COUNT

This is the total number of times that a page allocation request for uploading data using the direct-path option failed due to insufficient memory, accumulated over past executions.

V$DISKTBL_INFO

The view displays information about disk tables.

Column name	Type	Description
TABLESPACE_ID	SMALLINT	The tablespace identifier
TABLE_OID	BIGINT	The table object identifier
DISK_TOTAL_PAGE_CNT	BIGINT	The total number of pages in a table
DISK_PAGE_CNT	BIGINT	The number of pages containing data in a table
SEG_PID	INTEGER	The page identifier of a segment of a table
META_PAGE	INTEGER	This column has been deprecated
FST_EXTRID	BIGINT	The RID of the first extent in a table
LST_EXTRID	BIGINT	The RID of the last extent in a table
PCTFREE	SMALLINT	See SYS_TABLES_
PCTUSED	SMALLINT	See SYS_TABLES_
INITRANS	SMALLINT	The initial number of transactions that can be simultaneously processed in one page
MAXTRANS	SMALLINT	The maximum number of transactions that can be simultaneously processed in one page
INITEXTENTS	INTEGER	The initial number of extents when a table is created
NEXTEXTENTS	INTEGER	The number of extents that can be allocated when a table is expanded
MINEXTENTS	INTEGER	The minimum number of extents in a table
MAXEXTENTS	INTEGER	The maximum number of extents in a table
COMPRESSED_LOGGING	INTEGER	Whether to compress a log for a table
IS_CONSISTENT	INTEGER	Whether an index is consistent

To display a view together with the name of the table on which it is based, use a query to join the performance view with a meta table as follows:

```
SELECT A.TABLE_NAME,
    B.DISK_PAGE_CNT,
    B.PCTFREE,
    B.PCTUSED
FROM SYSTEM_.SYS_TABLES_ A, V$DISKTBL_INFO B
WHERE A.TABLE_OID = B.TABLE_OID;
```

Column Information

PCTFREE

Please refer to the description of the corresponding column in the SYS_TABLES_ description.

PCTUSED

Please refer to the description of the corresponding column in the SYS_TABLES_ description.

INITRANS

This is the initial number of transactions that can be processed simultaneously in one table page.

MAXTRANS

This is the maximum number of transactions that can be processed simultaneously in one table page.

INITEXTENTS

This is the initial number of extents when a table segment is created.

NEXTEXTENTS

This is the number of additional extents that will be allocated when the size of a table segment is increased.

MINEXTENTS

This is the minimum number of extents in a table segment.

MAXEXTENTS

This is the maximum number of extents in a table segment.

V$DISK_BTREE_HEADER

This view displays information about the header of a disk BTREE index.

Column name	Type	Description
INDEX_NAME	CHAR(128)	The index name
INDEX_ID	INTEGER	The index identifier

Column name	Type	Description
INDEX_TBS_ID	INTEGER	The tablespace in which the index is saved
TABLE_TBS_ID	INTEGER	The tablespace in which the table is saved
IS_UNIQUE	CHAR(1)	Whether an index is a unique key index
COLLENINFO_LIST	CHAR(64)	A list of the sizes of the values in the index
IS_CONSISTENT	CHAR(1)	Whether an index is consistent
IS_CREATED_WITH_LOGGING	CHAR(1)	Whether the LOGGING option was specified at the time the index was created
IS_CREATED_WITH_FORCE	CHAR(1)	Whether the NOLOGGING FORCE or NOLOGGING NOFORCE option was specified at the time the index was created
COMPLETION_LSN_LFG_ID	INTEGER	The log group identifier when the index was created
COMPLETION_LSN_FILE_NO	INTEGER	The log file number when the index was created
COMPLETION_LSN_FILE_OFFSET	INTEGER	The log file offset when the index was created
INIT_TRANS	SMALLINT	The initial number of transactions that can be simultaneously processed in a single index node
MAX_TRANS	SMALLINT	The maximum number of transactions that can be simultaneously processed in a single index node
FREE_NODE_HEAD	INTEGER	The ID of the first page in a free node
FREE_NODE_CNT	BIGINT	The number of pages in a free node list
INITEXTENTS	INTEGER	The initial number of extents when the index was created.
NEXTEXTENTS	INTEGER	The number of extents to be allocated when the index is increased in size
MINEXTENTS	INTEGER	The minimum number of extents in the index segment
MAXEXTENTS	INTEGER	The maximum number of extents in the index segment

Column Information

INDEX_NAME

This is the name of the index.

INDEX_ID

This displays the identifier, unique in the system, of the index.

INDEX_TBS_ID

This is the identifier of the tablespace in which the index is saved.

TABLE_TBS_ID

This is the identifier of the tablespace containing the table that is connected to the corresponding index.

IS_UNIQUE

This indicates whether the index is a unique key index. It is set to 'T' for a unique key index, and to 'F' for a duplicate key index.

- T: Unique key index
- F: Duplicate key index

COLLENINFO_LIST

This is a list of the sizes of the values in the index. The list is expressed as a comma-delimited string. The size of a variable length column is expressed as '?'. The size of a key can be inferred based on this list.

```
iSQL> CREATE TABLE D3(I1 SMALLINT, I2 INTEGER, I3 VARCHAR(10), I4 DATE) TABLESPACE
SYS_TBS_DISK_DATA;
Create success.
iSQL> CREATE INDEX D3X ON D3(I4,I3,I2,I1);
Create success.
iSQL> SELECT COLLENINFO_LIST FROM V$DISK_BTREE_HEADER WHERE INDEX_NAME='D3X';
COLLENINFO_LIST
------------------------------------------------------------------
8,?,4,2
1 row selected.
```

IS_CONSISTENT

This indicates whether the index is consistent. It is usually set to 'T'. It may be set to 'F' when an index is created with NOLOGGING or NOFORCE.

- T: Normal
- F: Abnormal

IS_CREATED_WITH_LOGGING

This indicates whether the LOGGING option was specified at the time that the index was created.

IS_CREATED_WITH_FORCE

This value indicates whether the NOLOGGING FORCE or NOLOGGING NOFORCE option was specified at the time that the index was created.

COMPLETION_LSN_FILE_NO

This is the log file number that was current at the time that the index was created.

COMPLETION_LSN_FILE_OFFSET

This is the log file offset that was current at the time that the index was created.

INIT_TRANS

This is the initial number of transactions that can simultaneously access a single index node (page) for an INSERT, UPDATE or DELETE operation.

MAX_TRANS

This is the maximum number of transactions that can simultaneously access a single index node (page) for an INSERT, UPDATE or DELETE operation.

FREE_NODE_HEAD

A FREE_NODE_HEAD shows the first page of a free node list within an index, a FREE NODE being a node in which a delete mark has been set for all keys therein.

FREE_NODE_CNT

This is the total number of FREE NODEs in an index.

INITEXTENTS

This is the initial number of extents, which is specified at the time that an index segment is created.

NEXTEXTENTS

This is the number of extents to be allocated when the size of an index segment is increased.

MINEXTENTS

This is the minimum number of extents in an index segment.

MAXEXTENTS

This is the maximum number of extents in an index segment.

V$DISK_TEMP_INFO

This performance view shows the usage information of the entire disk temporary table.

Column name	Type	Description
NAME	CHAR(32)	The minimum value name for memory
VALUE	CHAR(32)	The memory minimum value
UNIT	CHAR(32)	The unit

Column Information

VALUE

Indicates the minimum amount of memory required to sort memory required to sort disk temporary tables in memory since the server was started.

V$DISK_TEMP_STAT

This view indicates the current memory usage of each disk temporary table. The statistics is collected if the value set is greater than the value specified in TEMP_STATS_WATCH_TIME property.

Column name	Type	Description
TBS_ID	INTEGER	The identifier of the tablespace
TRANSACTION_ID	INTEGER	The transaction identifier
CONSUME_TIME	INTEGER	The execution time of disk temporary table
READ_COUNT	BIGINT	The number of IOs that reads data
WRITE_COUNT	BIGINT	The number of IOs that stores data.
WRITE_PAGE_COUNT	BIGINT	The total number of pages stored to disk
ALLOC_WAIT_COUNT	BIGINT	The total number of times waiting to allocate memory space.
WRITE_WAIT_COUNT	BIGINT	The total number of times waiting to store to disk.
QUEUE_WAIT_COUNT	BIGINT	The number of times to wait for a queue entry.
WORK_AREA_SIZE	BIGINT	The memory size that disk temporary tables use.
DISK_USAGE	BIGINT	The size of data space stored in disk.

Column Information

TBS_ID

This is the identifier of tablespace using disk temporary tables.

TRANSACTION_ID

This is the identifier of a transaction using disk temporary tables.

CONSUME_TIME

This shows the execution time if the execution time of a disk temporary table exceeds the time specified in TEMP_STATS_WATCH_TIME property.

READ_COUNT

This is the number of READ IO occurrences in order to read data on disk.

WRITE_COUNT

This is the number of WRITE IO occurrences in order to store data on disk.

WRITE_PAGE_COUNT

This is the total number of pages that disk temporary tables are stored into disk.

ALLOC_WAIT_COUNT

This is the number of times waiting to allocate memory space for hash sorting.

WRITE_WAIT_COUNT

This is the number of times waiting to store data on disk.

QUEUE_WAIT_COUNT

This is the number of times waiting to be queued to store data on disk.

WORK_AREA_SIZE

This is the space used for hash sorting in memory.

DISK_USAGE

This is the size of space that the disk temporary tables are stored to disk.

V$DISK_UNDO_USAGE

This view displays the amount of undo tablespace on disk that is currently being used.

Column name	Type	Description
TX_EXT_CNT	BIGINT	The number of extents in all transaction segments
USED_EXT_CNT	BIGINT	The number of extents currently being used in undo segments

Column name	Type	Description
UNSTEALABLE_EXT_CNT	BIGINT	The number of extents that cannot be stolen by other undo segments (when a segment does not have enough extents, it can take extents from other undo segments)
REUSABLE_EXT_CNT	BIGINT	The number of extents that can be reused
TOTAL_EXT_CNT	BIGINT	The total number of extents in the undo tablespace

Column Information

TX_EXT_CNT

This is the number of extents in all transaction segments. These extents cannot be used in undo segments.

USED_EXT_CNT

This is the number of extents currently used in undo segments. Because these extents are currently being used, they cannot be reused by subseqent tasks.

REUSABLE_EXT_CNT

This is the number of extents that can be reused because they contain undo records that are no longer necessary.

V$EVENT_NAME

This displays information about various wait events for which an Altibase server is waiting.

Column name	Type	Description
EVENT_ID	INTEGER	The identifier of a wait event
NAME	VARCHAR(128)	The name of the wait event
WAIT_CLASS_ID	INTEGER	The identifier of a wait class
WAIT_CLASS	VARCHAR(128)	The name of the wait class

Column Information

EVENT_ID

This is the identifier of the wait event.

NAME

This is the name of the wait event. The identifiers, names and corresponding descriptions are given in the following table.

EVENT_ID	NAME	Description

EVENT_ID	NAME	Description
0	latch: buffer busy waits	A wait to access a block being changed by another session
1	latch: drdb B-tree index SMO	A wait caused by a session that is executing a Structure Modification Operation (SMO) of a B-tree index
2	latch: drdb B-tree index SMO by other session	A wait until the completion of an SMO of a B-tree index by another session
3	latch: drdb R-tree index SMO	A wait caused by a session that is executing an SMO of an R-tree index
4	db file multi page read	A wait caused by a session that is waiting for the completion of a request to read multiple pages
5	db file single page read	A wait caused by a session that is waiting for the completion of a request to read a single page
6	db file single page write	A wait until a free BCB is obtained before an LRU flush can be executed
7	enq: TX – row lock contention, data row	A wait to place a lock on a row so that it can be updated
8	enq: TX – allocate TXSEG entry	A wait to assign a transaction segment entry
9	latch free: drdb file i/o	A wait to obtain a file latch in order to perform read/write I/O on a disk file
10	latch free: drdb tbs list	A wait to obtain a hash latch on a tablespace being used by another thread

EVENT_ID	NAME	Description
11	latch free: drdb tbs creation	A wait caused by a session that is attempting to create a file when a tablespace is created
12	latch free: disk page list entry	A wait to obtain a latch on a disk page list being used by another thread
13	latch free: drdb transaction segment freelist	A wait for a transaction segment free list
14	latch free: drdb LRU list	A wait for an LRU list in the buffer pool
15	latch free: drdb prepare list	A wait for a prepare list in the buffer pool
16	latch free: drdb prepare list wait	A wait until a BCB has been added to a prepare list in the buffer pool
17	latch free: drdb flush list	A wait for a flush list in the buffer pool
18	latch free: drdb checkpoint list	A wait for a checkpoint list in the buffer pool
19	latch free: drdb buffer flusher min recovery LSN	A wait for a latch for concurrency control of a Recovery LSN of the buffer pool flusher

EVENT_ID	NAME	Description
20	latch free: drdb buffer flush manager req job	A wait for a latch for concurrency control of a flush job of the buffer pool
21	latch free: drdb buffer bcb mutex	A wait for a latch for concurrency control of a BCB of the buffer pool
22	latch free: drdb buffer bcb read io mutex	A wait for a latch on a BCB of the buffer pool for page loading
23	latch free: drdb buffer buffer manager expand mutex	A wait for expansion of the buffer pool
24	latch free: drdb buffer hash mutex	A wait for a buffer pool hash
25	latch free: plan cache LRU List mutex	A wait to obtain a latch on an LRU list in a plan cache when adding, moving, or removing a plan from the list.
26	latch free: statement list mutex	A wait to obtain a latch on a statement list when adding, moving, or removing a statement from the list.
27	latch free: others	A wait to obtain a latch on anything being used by another thread that was not mentioned above

EVENT_ID	NAME	Description
28	replication before commit	In EAGER mode, this is the local server waiting to commit a transaction until all of the XLogs corresponding to statements that preceded the COMMIT statement have been replayed on the remote server. (Please refer to the description of EAGER mode in the Altibase *Replication Manual*.)
29	replication after commit	In EAGER mode, this is the local server waiting to commit a transaction until the XLog corresponding to the COMMIT statement has been sent to the remote server. (Please refer to the description of EAGER mode in the *Replication Manual*.)
30	no wait event	No wait event exists

WAIT_CLASS_ID

Wait events are conceptually grouped into broadly defined wait classes. For more detailed information about these wait classes, please refer to V$WAIT_CLASS_NAME.

WAIT_CLASS

This is the identifier of the class of a wait event. For more detailed information about wait class identifiers, please refer to V$WAIT_CLASS_NAME.

V$EXTPROC_AGENT

This is the meta table that contains information about the Agent Process created for the execution of external procedures.

Column name	Type	Description
SID	INTEGER	The identifier of the session which created the Agent Process
PID	INTEGER	The Pid of the Agent Process
SOCK_FILE	VARCHAR(64)	The socket path for communication between processes
CREATED	INTEGER	The date and time at which the Agent Process was created
LAST_SEND	INTEGER	The data and time at which the Agent Process returned the result for the last time
LAST_RECV	INTEGER	The date and time at which the Agent Process received the call message for the last time
STATE	VARCHAR(11)	The status of the Agent Process

Column Information

SID

This is the identifier of the session which created the Agent Process. The Agent Process is subordinate to the session

PID

This is the process ID of the Agent Process.

SOCK_FILE

This is the path of a socket used for communicaiton between processes.

CREATED

This is the date and time at which the Agent Process was created.

LAST_SEND

This is the date and time at which the Agent Process returned a result to the server session that called an external procedure for the last time

LAST_RECV

This is the date and time at which the Agent Process received a call message from the server session for the last time.

STATE

This is the status of the Agent Process and it takes one of the following values.

- INITIALIZED: Is newly created and waiting for a call.

- RUNNING: The external procedure is being executed.

- STOPPED: Execution of the external procedure is complete.

- FAILED: Has been either terminated abnormally, or the Agent Process has been terminated already.

V$FILESTAT

This view displays cumulative statistical information about I/O on individual disk files since the system was started. These statistics can be used to determine which data files are hot spots.

Column name	Type	Description
SPACEID	INTEGER	The tablespace identifier
FILEID	INTEGER	The data file identifier
PHYRDS	BIGINT	The number of physical read I/O operations that have been conducted

Column name	Type	Description
PHYWRTS	BIGINT	The number of physical write I/O operations that have occurred
PHYBLKRD	BIGINT	The number of physical read I/O operations that have been conducted
PHYBLKWRT	BIGINT	The number of pages that have been physically written to disk
SINGLEBLKRDS	BIGINT	The number of read operations that have taken place on single pages
READTIM	DOUBLE	The total time (in milliseconds) spent on read I/O operations
WRITETIM	DOUBLE	The total time (in milliseconds) spent on write operations
SINGLEBLKRDTIM	DOUBLE	The total time taken to read a single page (in milliseconds)
AVGIOTIM	DOUBLE	The average time (in milliseconds) per I/O operation
LSTIOTIM	DOUBLE	The time (in milliseconds) spent performing the most recent I/O operation
MINIOTIM	DOUBLE	The shortest time (in milliseconds) spent on a single I/O operation
MAXIORTM	DOUBLE	The longest time (in milliseconds) spent performing a single read operation
MAXIOWTM	DOUBLE	The longest time (in milliseconds) spent performing a single write operation

Column Information

SPACEID

This is the identifier of the tablespace.

FILEID

This is the identifier of the data file.

PHYRDS

This is the total number of physical read I/O operations that have been performed.

PHYWRTS

This is the total number of physical write operations that have been performed.

PHYBLKRD

This is the total number of pages that have been opened for physical reading.

PHYBLKWRT

This is the total number of pages that have been physically written to disk.

SINGLEBLKRDS

This is the total number of read I/O operations that have been performed on single pages.

READTIM

This is the total time (in milliseconds) spent performing read I/O operations.

WRITETIM

This is the total time (in milliseconds) spent performing write I/O operations.

SINGLEBLKRDTIM

This is the total amount of time (in milliseconds) spent performing read I/O operations on single pages.

AVGIOTIM

This is the average time (in milliseconds) spent performing a single I/O operation.

LSTIOTIM

This is the time (in milliseconds) spent performing the most recent I/O operation.

MINIOTIM

This is the minimum time (in milliseconds) spent performing a single I/O operation.

MAXIORTM

This is the maximum time (in milliseconds) spent performing a single read I/O operation.

MAXIOWTM

This is the maximum time (in milliseconds) spent performing a single write I/O operation.

V$FLUSHER

This view displays information about flushing tasks.

Column name	Type	Description
ID	INTEGER	This is the identifier of the flusher
ALIVE	INTEGER	This indicates whether the flusher is currently active.

Column name	Type	Description
CURRENT_JOB	INTEGER	Current job 1: replacement flushing is underway 2: checkpoint flushing is underway 3: an object is being flushed
DOING_IO	INTEGER	This indicates whether the flusher is performing disk I/O.
INIOB_COUNT	INTEGER	This is the number of times that an internal buffer has been directly accessed in order to save contents to be flushed therein.
REPLACE_FLUSH_JOBS	BIGINT	This is the cumulative number of replacement flushing tasks that have been completed
REPLACE_FLUSH_PAGES	BIGINT	This is the cumulative number of pages that have been written to disk by replacement flushing.
REPLACE_SKIP_PAGES	BIGINT	This is the cumulative number of pages for which flushing was canceled during replacement flushing.
CHECKPOINT_FLUSH_JOBS	BIGINT	This is the cumulative number of checkpoint flushing tasks that have been completed.
CHECKPOINT_FLUSH_PAGES	BIGINT	This is the cumulative number of pages that have been written to disk by checkpoint flushing.
CHECKPOINT_SKIP_PAGES	BIGINT	This is the cumulative number of pages for which flushing was canceled during checkpoint flushing.
OBJECT_FLUSH_JOBS	BIGINT	This is the cumulative number of times that object flushing has been performed.
OBJECT_FLUSH_PAGES	BIGINT	This is the cumulative number of pages that have been written to disk by object flushing.
OBJECT_SKIP_PAGES	BIGINT	This is the cumulative number of pages for which flushing was canceled during object flushing.
LAST_SLEEP_SEC	INTEGER	This is the length of time that the flusher has slept after having completed all of its tasks.
TIMEOUT	BIGINT	This is the number of times that a sleeping flusher has woken up in order to check whether it has any tasks

Column name	Type	Description
SIGNALED	BIGINT	This is the number of times that the flusher has been woken up by a signal from Altibase.
TOTAL_SLEEP_SEC	BIGINT	This is the total length of time that the flusher has slept.
TOTAL_FLUSH_PAGES	BIGINT	The cumulative number of pages that have been flushed
TOTAL_LOG_SYNC_USEC	BIGINT	The cumulative amount of time taken to write buffer-resident redo logs to disk
TOTAL_DW_USEC	BIGINT	The cumulative amount of time to taken write the contents of doublewrite buffers to disk
TOTAL_WRITE_USEC	BIGINT	The cumulative amount of time to taken to write data pages to data files
TOTAL_SYNC_USEC	BIGINT	The cumulative amount of time to taken to forcibly flush data pages to disk
TOTAL_FLUSH_TEMP_PAGES	BIGINT	The cumulative number of temporary pages that have been flushed
TOTAL_TEMP_WRITE_USEC	BIGINT	The cumulative amount of time to taken to write temporary pages to temporary files
TOTAL_CALC_CHECKSUM_USEC	BIGINT	The cumulative amount of time to taken to perform checksum calculations
DB_WRITE_PERF	DOUBLE	The average number of bytes that are written per second when writing data pages to data files
TEMP_WRITE_PERF	DOUBLE	The average number of bytes that are written per second when writing temporary pages to temporary files

Column Information

ID

This is the identifier of the flusher. A newly created identifier cannot be a duplicate of an existing identifier.

ALIVE

This indicates whether the flusher is currently active. Individual flushers can be started or stopped using DCL statements.

CURRENT_JOB

This indicates the type of job that the flusher is currently performing.

- A value of 1 indicates that the flusher is performing replacement flushing. The purpose of replacement flushing is to flush buffers that have not been accessed for a long time so that they can be replaced.
- A value of 2 indicates that the flusher is performing checkpoint flushing. The purpose of checkpoint flushing is to flush the buffer that has not been flushed for the longest time in order to reduce the amount of time required to perform checkpointing.
- A value of 3 indicates that the flusher is performing object flushing on a particular object, such as an index, table, segment, etc

DOING_IO

This indicates whether the flusher is currently performing disk I/O in order to fulfill its current task.

INIOB_COUNT

In order to save pages to disk, their contents are saved in an internal buffer (IOB). This value indicates the number of times that this internal buffer has been directly accessed in order to save contents to be flushed therein.

REPLACE_FLUSH_JOBS

This is the cumulative number of replacement flush opeartions performed.

REPLACE_FLUSH_PAGES

This is the cumulative number of pages that have been written to disk in the course of performing replacement flushing tasks.

REPLACE_SKIP_PAGES

This is the cumulative number of pages for which a flushing task was canceled during replacement flushing. Such cancellation can occur either according to some policy or in the interests of efficiency.

CHECKPOINT_FLUSH_JOBS

This the cumulative number of checkpoint flush operations.

CHECKPOINT_FLUSH_PAGES

This is the cumulative number of pages that have been written to disk in the course of performing checkpoint flushing tasks.

CHECKPOINT_SKIP_PAGES

This is the cumulative number of pages for which a flushing task was canceled during checkpoint flushing. Such cancellation can occur either according to some policy or in the interests of efficiency.

OBJECT_FLUSH_JOBS

The is the cumulative number of times that an object was flushed.

OBJECT_FLUSH_PAGES

This is the cumulative number of pages that have been written to disk in the course of performing object flushing tasks.

OBJECT_SKIP_PAGES

This is the cumulative number of pages for which a flushing task was canceled during object flushing. Such cancellation can occur either according to some policy or in the interests of efficiency.

LAST_SLEEP_SEC

This is the length of time the flusher has most recently slept after having completed all of its tasks.

TIMEOUT

Flushers that have no tasks and thus go to sleep are required to wake up at regular intervals to check whether they have work to do. This is the number of times that this has occurred.

SIGNALED

In order to improve the performance with which some task is performed, Altibase can signal a sleeping flusher and wake it up. This value is the number of times that the flusher has been woken up by such a signal.

TOTAL_SLEEP_SEC

This is the total length of time that the flusher has slept because the flusher did not have any work to do.

TOTAL_FLUSH_PAGES

This is the cumulative number of pages that have been flushed in the course of checkpoint flushing or replacement flushing.

TOTAL_LOG_SYNC_USEC

When data pages are flushed, redo logs must first be written to disk using the WAL (Write Ahead Logging) method. This is the cumulative amount of time taken to write redo logs to disk.

TOTAL_DW_USEC

This is the cumulative amount of time taken to write the contents of doublewrite buffers to disk. In so-called "doublewrite", pages are first written to DW ("doublewrite") files, i.e. the disk-resident doublewrite buffer. Once this process is complete, the pages are then written to data files in the usual location. If the operating system crashes during the process of writing pages to data files, or if these data files become corrupted, it will be possible to perform data recovery using the uncorrupted copies of the pages in the doublewrite buffer.

TOTAL_WRITE_USEC

This is the cumulative amount of time taken to write data pages to data files. This value does not include the amount of time spent flushing data to disk.

TOTAL_SYNC_USEC

This is the cumulative amount of time spent forcibly flushing data to disk.

TOTAL_FLUSH_TEMP_PAGES

This is the cumulative number of temporary pages that have been flushed. (Temporary pages are used for storing temporary tables, which are used for sort operations and hash joins.)

TOTAL_TEMP_WRITE_USEC

This is the amount of time spent writing temporary pages to temporary files.

TOTAL_CALC_CHECKSUM_USEC

This is the amount of time taken to calculate checksums, which are used to determine whether pages are corrupt.

DB_WRITE_PERF

This is the average number of bytes that are written per second (in kB/sec) when data pages are written to data files.

TEMP_WRITE_PERF

This is the average number of bytes that are written per second (kB/sec) when temporary pages are written to temporary files.

V$FLUSHINFO

This view displays buffer flush information.

Column name	Type	Description
LOW_FLUSH_LENGTH	INTEGER	The minimum length of the flush list above which replacement flushing can occur
HIGH_FLUSH_LENGTH	INTEGER	The flush list length at which the flusher ignores REPLACE_FLUSH_COUNT and flushes all the buffers in the flush list.
LOW_PREPARE_LENGTH	INTEGER	The threshold length of the prepare list that can cause replacement flushing. Replacement flushing occurs when the prepare list is shorter than this length.
CHECKPOINT_FLUSH_COUNT	BIGINT	The number of buffers to be flushed when checkpoint flushing occurs.

Column name	Type	Description
FAST_START_IO_TARGET	BIGINT	The number of dirty pages that will not be flushed when checkpoint flushing occurs
FAST_START_LOGFILE_TARGET	INTEGER	The number of log files that will not be flushed when checkpoint flushing occurs
REQ_JOB_COUNT	INTEGER	The number of tasks currently registered for the flush manager

Column Information

LOW_FLUSH_LENGTH

This is the minimum length of the flush list above which replacement flushing can occur.

HIGH_FLUSH_LENGTH

This is the flush list length at which the flusher ignores REPLACE_FLUSH_COUNT and flushes all the buffers in the flush list.

LOW_PREPARE_LENGTH

This is the threshold length of the prepare list. Replacement flushing occurs if the length of a prepare list drops below this length.

CHECKPOINT_FLUSH_COUNT

This is the number of buffers that will be flushed when checkpoint flushing is performed.

FAST_START_IO_TARGET

This is the number of dirty pages that are not flushed when checkpoint flushing occurs.

FAST_START_LOGFILE_TARGET

This is the number of log files that are not flushed when checkpoint flushing occurs. These are the most recently created log files.

REQ_JOB_COUNT

This is the number of jobs registered in the flush manager.

V$INDEX

This view shows information about the indexes that currently exist in the database:

Column name	Type	Description
TABLE_OID	BIGINT	The object identifier of the table header
INDEX_SEG_PID	INTEGER	The page identifier of a segment header in the case of a disk index

Column name	Type	Description
INDEX_ID	INTEGER	The identifier of the index
INDEXTYPE	VARCHAR(7)	An indicator that identifies whether the index is a primary key or a standard index

Column Information

TABLE_OID

This is the object identifier of the table for which the index was created, and stores the physical location of the header, which contains the table information.

INDEXTYPE

This indicates whether the index is used as a primary key or as a normal index.

- PRIMARY: The index is used as primary key.
- NORMAL: The index is used as normal one

V$INSTANCE

This view displays information about an Altibase database, the amount of time it took to start up, and the amount of time that has elapsed since startup.

Column name	Type	Description
STARTUP_PHASE	VARCHAR(13)	The current startup phase
STARTUP_TIME_SEC	BIGINT	The system time at which the system was started (in seconds).
WORKING_TIME_SEC	BIGINT	The amount of time that has elapsed from startup to the present

V$INTERNAL_SESSION

This view displays information about a session created in the DBMS_CONCURRENT_EXEC package. For further information, please refer to V$SESSION.

Column name	Type	Description
ID	BIGINT	The session ID
TRANS_ID	BIGINT	The ID of the transaction that is currently being executed in the session
QUERY_TIME_LIMIT	BIGINT	The amount of time by which a query running in the session exceeded the specified time limit

Column name	Type	Description
DDL_TIME_LIMIT	BIGINT	The amount of time by which a DDL statement running in the session exceeded the specified time limit
FETCH_TIME_LIMIT	BIGINT	The amount of time by which a fetch operation running in the session exceeded the specified time limit
UTRANS_TIME_LIMIT	BIGINT	The amount of time by which an update transaction running in the session exceeded the specified time limit
IDLE_TIME_LIMIT	BIGINT	The amount of time by which the current session exceeded the specified idle time limit
IDLE_START_TIME	INTEGER	The time at which the session becomes inactive (idle)
ACTIVE_FLAG	INTEGER	The active transaction flag
OPENED_STMT_COUNT	INTEGER	The number of statements being executed in the session
DB_USERNAME	VARCHAR(128)	The database user name
DB_USERID	INTEGER	The database user ID
DEFAULT_TBSID	BIGINT	The ID of the user's default tablespace
DEFAULT_TEMP_TBSID	BIGINT	The ID of the user's default temporary tablespace
SYSDBA_FLAG	INTEGER	Whether the session is connected as SYSDBA
AUTOCOMMIT_FLAG	INTEGER	The autocommit flag
SESSION_STATE	VARCHAR(13)	The session state
ISOLATION_LEVEL	INTEGER	The session isolation level
REPLICATION_MODE	INTEGER	The replication mode
TRANSACTION_MODE	INTEGER	The transaction mode
COMMIT_WRITE_WAIT_MODE	INTEGER	See below
OPTIMIZER_MODE	INTEGER	The optimizer mode

Column name	Type	Description
HEADER_DISPLAY_MODE	INTEGER	Indicates whether only the column names are output, or whether the table names are output along with the column names when the results of a SELECT query are output. 0: The table names are displayed along with the column names. 1: Only the column names are output.
CURRENT_STMT_ID	INTEGER	The ID of the statement that is currently being executed
STACK_SIZE	INTEGER	The size of the stack for query processing (Unit: bytes)
DEFAULT_DATE_FORMAT	VARCHAR(64)	The default date format (e.g. DD-MON-RRRR)
TRX_UPDATE_MAX_LOGSIZE	BIGINT	The maximum size of the DML log (Unit: bytes)
PARALLE_DML_MODE	INTEGER	Deprecated
LOGIN_TIME	INTEGER	The time at which the client was logged in
FAILOVER_SOURCE	VARCHAR(256)	Information about the connection when a failover occurred
NLS_TERRITORY	VARCHAR(40)	The territory name of the session
NLS_ISO_CURRENCY	VARCHAR(40)	The ISO currency code of the session
NLS_CURRENCY	VARCHAR(10)	The local currency symbol of the session
NLS_NUMERIC_CHARACTERS	VARCHAR(2)	The group separator and decimal character of the session
TIME_ZONE	VARCHAR(40)	The territory name/abbreviation or UTC_OFFSET of the specified time zone for the session
LOB_CACHE_THRESHOLD	INTEGER	The value specified for the LOB_CACHE_THRESHOLD property
QUERY_REWRITE_ENABLE	VARCHAR(7)	The value specified for the QUERY_REWRITE_ENABLE property

Column Information

TRANS_ID

Indicates the transaction identifier currently running in the session. If no transaction is currently running, this value is -1.

ACTIVE_FLAG

If the session is executing a statement, the value is 1. If the session is merely connected or has committed/rolled back a transaction, the value is 0.

SYSDBA_FLAG

Indicates whether the connected session is in sysdba mode or not.

- 1: sysdba mode

AUTOCOMMIT_FLAG

Indicates whether or not the session is in AUTOCOMMIT mode.

- 0: NON-AUTOCOMMIT
- 1: AUTOCOMMIT

SESSION_STATE

STATE	Description
INIT	Waiting for the client to request
AUTH	Finished user authentication
SERVICE READY	Ready for service. (Unable to start a transaction. Only an XA session can have this state.)
SERVICE	Servicing
END	Terminated normally (If a transaction exists, it has been committed successfully.)
ROLLBACK	Terminated abnormally (If a transaction exists, it has been rolled back.) This state occurs if the client was disconnected or the server forcefully killed the session.
UNKNOWN	N/A

REPLICATION_MODE

The replicaiton mode.

- 0: DEFAULT
- 16: NONE

TRANSACTION_MODE

The transaction mode.

- 0: READ/WRITE
- 4: READ ONLY

COMMIT_WRITE_WAIT_MODE

- 0: when commiting, do not wait for the log to be written to disk
- 1: when commiting, wait for the log to be written to disk

OPTIMIZER_MODE

Indicates the optimization mode set for the session.

- 1: Rule based
- 0: Cost based

QUERY_REWRITE_ENABLE

Indicates the value set for the QUERY_REWRITE_ENABLE property in the session. Please refer to Chapter 2 for the QUERY_REWRITE_ENABLE property.

- FALSE: Disable function-based indexes when converting queries on the Altibase server
- TRUE: Enable function-based indexes when converting queries on the Altibase server.

V$LATCH

This view displays statistical information about the BCB latch of the buffer pool, including the number of attempts to obtain a latch on pages on which it is desired to perform read or write I/O, the number of latches that were successfully obtained immediately, and the number of failures to obtain a latch. These statistics are calculated separately for read and write latches.

Column name	Type	Description
SPACE_ID	INTEGER	The tablespace identifier
PAGE_ID	INTEGER	The page identifier
TRY_READ_LATCH	BIGINT	The number of attempts to obtain read latches
READ_SUCCESS_IMME	BIGINT	The number of immediate successes to obtain read latches
READ_MISS	BIGINT	The number of failures to obtain read latches
TRY_WRITE_LATCH	BIGINT	The number of attempts to obtain write latches
WRITE_SUCCESS_IMME	BIGINT	The number of immediate successes to obtain write latches
WRITE_MISS	BIGINT	The number of failures to obtain write latches

Column name	Type	Description
SLEEPS_CNT	BIGINT	The number of sleeps related to latch attempts

V$LIBRARY

This view provides informaiton of dynamically loaded library in C/C++ internal procedure. The user can check whether the desired library is properly loaded with the library information.

Column name	Type	Description
FILE_SPEC	CHAR(4000)	The path to dynamic library files
REFERENCE_COUNT	INTEGER	The number of internal procedures referencing dynba
FILE_SIZE	INTEGER	The file size of dynamic library (bytes)
CREATE_TIME	VARCHAR(48)	The time the dynamic library was created
OPEN_TIME	VARCHAR(48)	The time the dynamic library was loaded

Column Information

FILE_SPEC

This indicates the path of the dynamic library file pointed to by the library object. It is displayed as a relative path to the default path ($ALTIBASE_HOME/lib) where the library files are located.

REFERENCE_COUNT

This indicates the number of internal stored procedures or stored functions referencing a dynamic library.

FILE_SIZE

This indicates the size of a dynamic library file. (Unit: bytes)

CREATE_TIME

This indicates the date and time when the dynamic library was created. Receive and save from file information.

OPEN_TIME

This indicates the date and time when dynamic library was loaded.

V$LFG

This view provides statistical information to help database administrators monitor group commit activity. For more detailed information about each column, please refer to the commit section in this manual.

Column name	Type	Description

Column name	Type	Description
LFG_ID	INTEGER	The log file group identifier
CUR_WRITE_LF_NO	INTEGER	The log file number of the log file currently being written to
CUR_WRITE_LF_OFFSET	INTEGER	The offset of the log file currently being written to
LF_OPEN_COUNT	INTEGER	The number of open log files
LF_PREPARE_COUNT	INTEGER	The number of log files that have been created in advance
LF_PREPARE_WAIT_COUNT	INTEGER	The number of waits to switch to new log files
LST_PREPARE_LF_NO	INTEGER	The identifier of the most recently prepared log file
END_LSN_LFGID	INTEGER	Not used. (0)
END_LSN_FILE_NO	INTEGER	The file number portion of the LSN (Log Sequence Number) at which a REDO operation will start when Altibase is restarted
END_LSN_OFFSET	INTEGER	The offset within a LSN (Log Sequence Number) at which a REDO operation will start when Altibase is restarted
FIRST_DELETED_LOGFILE	INTEGER	The first log file that was deleted (inclusive)
LAST_DELETED_LOGFILE	INTEGER	The log file is the last log file that was deleted
RESET_LSN_LFGID	INTEGER	Not used. (0)
RESET_LSN_FILE_NO	INTEGER	The file number portion of the LSN (Log Sequence Number) used after database recovery
RESET_LSN_OFFSET	INTEGER	The offset of the LSN (Log Sequence Number) used after database recovery
UPDATE_TX_COUNT	INTEGER	The number of transactions in the LFG that are currently making changes to the database (only available for group commit)
GC_WAIT_COUNT	INTEGER	The number of waits for disk I/O (only available for group commit)
GC_ALREADY_SYNC_COUNT	INTEGER	The number of completed disk I/O operations (only available for group commit)
GC_REAL_SYNC_COUNT	INTEGER	The number of actual disk I/O operations that occurred during group commit

Column Information

LFG_ID

This is a unique log file group number of the value 0.

CUR_WRITE_LF_NO

This is the number of the log file currently being used to store logs.

CUR_WRITE_LF_OFFSET

This is the log file offset currently being used to store logs.

LF_OPEN_COUNT

This is the number of log files on disk that are open for use by Altibase.

LF_PREPARE_COUNT

This is the number of log files that have been created in advance (prepared) by the log file creation thread up to the present moment.

LF_PREPARE_WAIT_COUNT

When all of the prepared log files have been used, it is necessary to create new log files. This is the total number of waits for log files to be created in order to switch to a new log file.

If this value is large, setting the PREPARE_LOG_FILE_COUNT property to a higher value will help ensure that a sufficient number of log files is prepared in advance. For more information about PREPARE_LOG_FILE_COUNT, please refer to the *General Reference*.

LST_PREPARE_LF_NO

This is the number of the log file that was most recently prepared (created in advance) by the log file creation thread.

END_LSN_FILE_NO

This shows the number of the log file, which is part of the LSN (Log Sequence Number), at which REDO commences when the system is restarted. It can be guaranteed that REDO will definitely begin with a log having a greater LSN value than the one shown here.

END_LSN_OFFSET

This shows the offset within the log file, which is part of the LSN (Log Sequence Number), at which REDO commences when the system is restarted. It can be guaranteed that REDO will definitely begin with a log having a greater LSN value than the one shown here.

FIRST_DELETED_LOGFILE

This shows the number of the first of the log files that were classified as unnecessary and deleted during checkpointing. This means that the log file having this number was deleted during checkpointing.

LAST_DELETED_LOGFILE

This shows the last log files that were classified as unnecessary and deleted during checkpointing. This number means that the corresponding log file has been deleted during the checkpoint.

RESET_LSN_FILE_NO

RESET_LSN is the first LSN after the time point at which recovery was performed. RESET_LSN_FILE_NO is the log file number portion of RESET_LSN.

RESET_LSN_OFFSET

This shows the offset within the log file, and is a portion of RESET_LSN.

UPDATE_TX_COUNT

This returns, in real time, the number of transactions in the LFG that are currently making changes to the database.

GC_WAIT_COUNT

This shows the total number of times transactions in this LFG had to wait for disk I/O for group commit.

GC_ALREADY_SYNC_COUNT

During group commit, it is sometimes not necessary to perform disk I/O for some transactions, because the logs containing them have already been written to disk. This is the cumulative number of times this has occurred.

GC_REAL_SYNC_COUNT

This shows the number of actual disk I/O operations related to transactions in this LFG during group commit.

V$LOCK

This view displays information about lock nodes for all tables in the database at the current point in time.

Column name	Type	Description
LOCK_ITEM_TYPE	VARCHAR(7)	The type of object that is locked
TBS_ID	INTEGER	The tablespace identifier
TABLE_OID	BIGINT	The table object identifier
DBF_ID	BIGINT	The database file identifier
TRANS_ID	BIGINT	The transaction identifier
LOCK_DESC	VARCHAR(32)	A character string indicating the lock mode e.g.) IX, IS, X
LOCK_CNT	INTEGER	The number of locks for this lock node

Column name	Type	Description
IS_GRANT	BIGINT	Indicates whether the table is locked or is waiting to be locked

Column Information

LOCK_ITEM_TYPE

This indicates the type of object that is locked, and can have the following values:

Value	Description
NONE	Cannot have this value
TBS	Tablespace
TBL	Table
DBF	Database file
UNKNOWN	Unknown object type

V$LOCK_STATEMENT

This view displays information about statements that are holding or waiting to acquire locks.

Column name	Type	Description
SESSION_ID	INTEGER	The session identifier
ID	INTEGER	The statement identifier
TX_ID	BIGINT	The transaction identifier
QUERY	VARCHAR(16384)	The query statement
STATE	INTEGER	The state of the statement
BEGIN_FLAG	INTEGER	A flag indicating the beginning of the statement
LOCK_ITEM_TYPE	VARCHAR(7)	The type of object that is locked
TBS_ID	INTEGER	The transaction identifier
TABLE_OID	BIGINT	The table object identifier
DBF_ID	BIGINT	The database file identifier
LOCK_DESC	VARCHAR(32)	A character string indicating the lock mode e.g.) IX, IS, X
LOCK_CNT	INTEGER	The number of locks for the lock node

Column name	Type	Description
IS_GRANT	BIGINT	Indicates whether the table is locked or is waiting to be locked

V$LOG

This view displays information about log anchors.

Column name	Type	Description
BEGIN_CHKPT_LFGID	INTEGER	Not used(0)
BEGIN_CHKPT_FILE_NO	INTEGER	The log file number of the checkpoint start log of the most recently executed checkpoint
BEGIN_CHKPT_FILE_OFFSET	INTEGER	The log offset of the checkpoint start log of the most recently executed checkpoint
END_CHKPT_LFGID	INTEGER	Not used(0)
END_CHKPT_FILE_NO	INTEGER	The log file number of the checkpoint end log of the most recently executed checkpoint
END_CHKPT_FILE_OFFSET	INTEGER	The log offset of the checkpoint end log of the most recently executed checkpoint
SERVER_STATUS	VARCHAR(15)	A character string indicating the status of the server
ARCHIVELOG_MODE	VARCHAR(12)	A character string indicating the status of database archive mode
TRANSACTION_SEGMENT_COUNT	INTEGER	The number of transaction segments to be created in the undo tablespace
OLDEST_LFGID	INTEGER	Not used(0)
OLDEST_LOGFILE_NO	INTEGER	When restart recovery is performed, the log file number from which disk-related redo will begin
OLDEST_LOGFILE_OFFSET	INTEGER	When restart recovery is performed, the log file offset from which disk-related redo will begin

Column Information

SERVER_STATUS

This is the status of the server.

- SERVER SHUTDOWN: The server has been shut down.
- SERVER STARTED: The server is running.

ARCHIVELOG_MODE

This indicates whether Archivelog mode is enabled for the database.

- ARCHIVE: In this mode, unnecessary log files are stored in an extra directory for use in performing media recovery.
- NOARCHIVE: In this mode, unnecessary log files are deleted.

V$LOCK_WAIT

This view shows wait information between transactions that are executed on the system.

Column name	Type	Description
TRANS_ID	BIGINT	The identifier of the waiting transaction
WAIT_FOR_TRANS_ID	BIGINT	The identifier of the transaction being waited for

Column Information

TRANS_ID

This is the identifier of the transaction that is currently waiting.

WAIT_FOR_TRANS_ID

This is the identifier of the transaction for which the transaction identified by TRANS_ID is waiting.

```
SQL> select * from v$lock_wait;
V$LOCK_WAIT.TRANS_ID  V$LOCK_WAIT.WAIT_FOR_TRANS_ID
-------------------------------------------
1216                             2208
5344                             2208
2 rows selected.
```

In the above example, transactions 1216 and 5344 are waiting for transaction 2208.

V$MEMGC

This view displays memory space recovery (that is, memory garbage collection) information.

Column name	Type	Description

Column name	Type	Description
GC_NAME	VARCHAR(128)	MEM_LOGICAL_AGER: Previous version index key slot release thread MEM_DELTHR: A thread that releases deleted records and supports pending operations such as DROP TABLE etc.
CURRSYSTEMVIEWSCN	VARCHAR(29)	The current system view SCN
MINMEMSCNINTXS	VARCHAR(29)	The lowest of the view SCNs for memory-related transactions
OLDESTTX	INTEGER	The identifier of the oldest transaction (the identifier of the transaction to which MINMEMSCNINTXS belongs)
SCNOFTAIL	VARCHAR(29)	The commit SCN of the tail in garbage collection OID list
IS_EMPTY_OIDLIST	BIGINT	Whether the garbage collection OID list is empty 0: empty 1: not empty
ADD_OID_CNT	BIGINT	The number of transactions that caused OIDs to be added for garbage collection management
GC_OID_CNT	BIGINT	The number of times OIDs are deleted for garbage collection
AGING_REQUEST_OID_CNT	BIGINT	The number of outdated versions of records for which deletion has been requested
AGING_PROCESSED_OID_CNT	BIGINT	The number of outdated versions of records that have been deleted
THREAD_COUNT	INTEGER	The number of garbage collection threads

Column Information

Because Altibase supports MVCC, multiple versions of a single record can exist. In other words, one record consists of a most recent version and a number of previous versions. For more detailed information on MVCC, please refer to the sections pertaining to Multi-Version Concurrency Control (MVCC) in both the *Altibase Administrator's Manual* and the *Altibase Getting Started Guide*.

AGING_REQUEST_OID_CNT

If 10 records are deleted in one transaction, which is then committed, there are now 10 outdated records that can be cleared to recover space. However, because ADD_OID_CNT is determined on the basis of transactions, it is incremented by 1. To remedy this, AGING_REQUEST_OID_CNT, which is determined on the basis of OIDs, is incremented by 10.

AGING_PROCESSED_OID_CNT

If the garbage collector (or ager) deletes 10 outdated versions of records from the same OID list, GC_OID_CNT is only incremented by 1 because it determined on the basis of lists. To remedy this, AGING_PROCESSED_OID_CNT, which is determined on the basis of OIDs, is incremented by 10.

THREAD_COUNT

This shows the number of garbage collection threads.

V$MEMSTAT

This view displays statistics about the memory being used by Altibase processes.

Column name	Type	Description
NAME	CHAR(64)	The name of the memory module
ALLOC_SIZE	BIGINT	The amount of memory being used by the module (in bytes)
ALLOC_COUNT	BIGINT	The number of units of memory that make up ALLOC_SIZE
MAX_TOTAL_SIZE	BIGINT	The maximum memory size of the module (in bytes)

Column Information

NAME

This is the name of the module being used by Altibase. This column contains the following memory modules.

Name	Description
Altiwrap	The memory used for Altiwrap
Async_IO_Manager	The memory that is used when asynchronous I/O occurs
Audit_Manager	The memory for Audit administrators
CatalogCache_Memory	Not currently used
Clock_Manager	The memory for the clock manager. The clock manager uses the CPU clock when it checks the system time.

Name	Description
CM_Buffer	The buffer memory used for communication (TCP, Unix domain Socket, IPC, and IPCDA)
CM_DataType	The memory that is used for sending and receiving large packets
CM_Interface	The memory used by CM Interface
CM_Multiplexing	The memory that is used for saving session information for communication
CM_NetworkInterface	The memory that is used for saving information about individual communication nodes
Condition_Variable	The memory used to manage condition variables for multithreaded control
DatabaseLink	The memory that is used by Database Link
Disaster_recovery	The memory used by disaster recovery
Disaster_recovery_Control	The memory used by the role manager in disaster recovery
Disaster_recovery_Executor	The memory used during disaster recovery
Disaster_recovery_Storage	Not currently used
Dynamic Module Loader	The memory used when loading shared libraries
External_Procedure	The memory used by external procedures
External_Procedure_Agent	The Memory used by the external procedure agent
Fixed_Table	The memory that is used for fixed tables
GIS_DataType	The memory that is used for handling GIS data
GIS_Disk_Index	The memory that is used for managing the Disk Spatial Index for GIS data
GIS_Function	The memory that is used for space-related calculations
GIS_TEMP_MEMORY	The memory that is used for creating R-tree indexes

Name	Description
IDU_MEM_OTHER	The memory that is used for creating R-tree indexes
Index_Memory	The memory used to manage index information
InMemoryRecovery_Memory	Not currently used
Latch	The administrative memory used by latch
Legacy_Transaction_Manager	The memory used to manage legacy transaction information
LOG_Memory	Not currently used
Main_Module_CDBC_CONDITIONBUF_MEMPOOL	Not currently used
Main_Module_CDBC_CURSORDATA_MEMPOOL	Not currently used
Main_Module_CDBC_MAIN	Not currently used
Main_Module_CDBC_QP	Not currently used
Main_Module_CDBC_STATE_MEMPOOL	Not currently used
Main_Module_Channel	The memory used by Altibase main module
Main_Module_DirectAttach	Not currently used
Main_Module_Distributed	The memory used for XA management
Main_Module_Queue	The memory that is used for queues
Main_Module_Thread	The memory that is used for managing threads
Main_Module_Utility	Not currently used
Mathematics	The memory that is used for various kinds of mathematical operations
MMAP	The memory allocated by mmap system call
Mutex	The memory that is used for managing mutexes
OS_Independent	Not currently used
Process_ThreadInfo	Not currently used
Profile_Manager	The memory that is used by the Profile Manager

Name	Description
Query_Binding	The memory that is used for binding host variables
Query_Common	Memory that is used for other purposes
Query_Common_Remote_Call	Not currently used
Query_Conversion	Not currently used
Query_DML	The memory that is used for executing DML statements
Query_Execute	The memory that is used when queries are executed
Query_Execute_Cache	The memory used for caching deterministic function results
Query_Result_Cache	The memory used to cache results
Query_Meta	The memory that is used to manage cached meta information, which is checked while the server is active
Query_Prepare	The memory that is used for preparing queries for execution
Query_PSM_Concurrent_Execute	The memory that is used for executing the DBMS_CONCURRENT_EXEC package
Query_PSM_Execute	The memory that is used for executing PSM (Persistent Stored Module)
Query_PSM_Node	The memory that is used for managing PSM array variables
Query_Sequence	The memory that is used for managing sequences
Query_Transaction	The memory that is used for executing triggers
Remote_Call_Client	Not currently used
Remote_Call_Server	Not currently used
Replication_Common	Not currently used
Replication_Control	The memory that is used by the Replication Manager

Name	Description
Replication_Data	The memory that is used for processing XLOGs
Replication_Executor	Not currently used
Replication_Met	The memory used by meta cache
Replication_Module_Property	Not currently used
Replication_Network	The memory that is used for communication for replication
Replication_Receiver	The memory that is used by the replication Receiver
Replication_Recovery	The memory that is used to perform recovery using replication
Replication_Sender	The memory that is used by the replication Sender
Replication_Storage	The memory that is used to apply XLOGs
Replication_Sync	The memory that is used for synchronization in replication
RESERVED	Allocated, but not allocated when using the TLSF memory manager
Socket_Manager	Not currently used
SQL Plan Cache Control	The memory that is used for the SQL Plan Cache
Storage_DataPort	Memory that is used for executing DataPort
Storage_Disk_Buffer	The memory that is used by the Disk Buffer Manager
Storage_Disk_Collection	The memory that is used for performing Direct-Path Insert and LOB calculations for disk tables
Storage_Disk_Datafile	The memory that is used for data file management tasks, such as creating I/O buffers and data file nodes
Storage_Disk_Index	The memory that is used for managing disk indexes

Name	Description
Storage_Disk_Page	The memory that is used for assigning disk LOB segment descriptors and disk table page list mutexes
Storage_Disk_Recovery	The memory that is used to ensure the consistency of a disk database
Storage_Disk_SecondaryBuffer	The memory used by secondary disk buffer manager
Storage_Global_Memory_Manager	Not currently used
Storage_Memory_Ager	The memory that is used for the garbage collector and the database recovery ("refining") thread
Storage_Memory_Collection	The memory that is used for managing records in memory tables
Storage_Memory_Index	The memory that is used for managing memory indexes
Storage_Memory_Interface	The memory that is used at the storage module interface level
Storage_Memory_Locking	The memory that is used for locking tables and tablespaces
Storage_Memory_Logical_Ager	Not currently used
Storage_Memory_Manager	The memory in which memory data are actually stored
Storage_Memory_Page	The memory that is used for managing memory pages
Storage_Memory_Recovery	The memory that is used to perform recovery
Storage_Memory_Recovery_Archive_Thread	Not currently used
Storage_Memory_Recovery_Chkpt_Thread	Not currently used
Storage_Memory_Recovery_LFG_Thread	Not currently used
Storage_Memory_Transaction	The memory that is used for managing transaction information
Storage_Memory_Utility	The memory that is used when the Storage Manager Tool is used

Name	Description
Storage_Tablespace	The memory that is used for managing and allocating tablespace nodes
SYSTEM	The memory allocated directly by the operating system using the malloc function
Tablespace Free Extent Pool	The memory that is used for managing free extent pools of tablespaces
Temp_Memory	The memory that is used when allocating temporary space
Thread_Stack	The memory used by the thread stack when the thread is created
Timer_Manager	The memory for the timer manager, which uses the timer thread when checking the system time
Transaction_DiskPage_Touched_List	The memory that is used for managing disk data pages that are affected by a transaction
Transaction_OID_List	The memory that is used for making the OID (object identifier) list of a memory database
Transaction_Private_Buffer	Not currently used
Transaction_Segment_Table	The memory that is used for managing Undo segments and Transaction Status segments
Transaction_Table	The memory that is used for assigning transaction objects
Transaction_Table_Info	The memory that is used for managing information about the tables changed by a transaction
Utility_Module	Not currently used
Volatile_Log_Buffer	Volatile Log Buffer memory
Volatile_Memory_Manager	The memory in which volatile memory data are stored
Volatile_Memory_Page	The memory that is used for managing volatile memory pages

Name	Description
WATCHDOG	Not currently used

ALLOC_SIZE

This indicates the memory usage of the module.

ALLOC_COUNT

This is the number of unit memories that make up ALLOC_SIZE in the module.

MAX_TOTAL_SIZE

This indicates the maximum memory size the module has.

V$MEMTBL_INFO

This view displays information about the status of memory tables.

Column name	Type	Description
TABLESPACE_ID	SMALLINT	The tablespace identifier
TABLE_OID	BIGINT	The table object identifier
MEM_PAGE_CNT	BIGINT	The number of pages containing fixed-length columns in the table
MEM_VAR_PAGE_CNT	BIGINT	The number of pages containing variable-length columns in the table
MEM_SLOT_PERPAGE	INTEGER	The number of slots that can be stored in a page containing fixed-length columns
MEM_SLOT_SIZE	BIGINT	The size of the fixed area in the table record
FIXED_ALLOC_MEM	DOUBLE	The amount of fixed memory area (in bytes) allocated to a table
FIXED_USED_MEM	BIGINT	The amount of fixed memory area (in bytes) actually being used by a table
VAR_ALLOC_MEM	DOUBLE	The amount of variable memory area (in bytes) allocated to a table
VAR_USED_MEM	BIGINT	The amount of variable memory area (in bytes) actually being used by a table
MEM_FIRST_PAGEID	BIGINT	The number of the first of the fixed-length pages in the table

Column name	Type	Description
STATEMENT_REBUILD_COUNT	BIGINT	The number of times a statement has been rebuilt
UNIQUE_VIOLATION_COUNT	BIGINT	The number of times a unique key violation has occurred
UPDATE_RETRY_COUNT	BIGINT	The number of times an update operation has been retried
DELETE_RETRY_COUNT	BIGINT	The number of times a delete operation has been retried
COMPRESSED_LOGGING	INTEGER	Indicates whether log compression is enabled or not
IS_CONSISTENT	INTEGER	Whether an table is consistent

To view this information together with the table name, join this view with the SYS_TABLES_ meta table and execute a query as follows:

```
SELECT A.TABLE_NAME,
    B.MEM_PAGE_CNT,
    B.MEM_SLOT_SIZE,
    B.MEM_FIRST_PAGEID
FROM    SYSTEM_.SYS_TABLES_ A, V$MEMTBL_INFO B
WHERE   A.TABLE_OID = B.TABLE_OID;
```

Column Information

TABLESPACE_ID

This is the identifier of the tablespace in which the current table is stored. The following tablespaces are created by default. Identifiers of new user-created tablespaces will have values greater than 4.

- 0: SYS_TBS_MEM_DIC
- 1: SYS_TBS_MEM_DATA
- 2: SYS_TBS_DISK_DATA
- 3: SYS_TBS_DISK_UNDO
- 4: SYS_TBS_DISK_TEMP

TABLE_OID

This is the default table object identifier, and indicates the physical location of the header that contains information about the table. This is only used internally by the system.

STATEMENT_REBUILD_COUNT

When the Prepare-Execute process is performed, a prepared statement is executed without being parsed, validated, or optimized. However, after the statement is prepared, if a DDL statement is executed on a query target object (a tablespace, table or index), the corresponding statement is automatically rebuilt when the statement is executed, and this value is incremented.

UNIQUE_VIOLATION_COUNT

This value is incremented when a unique key restriction is violated.

UPDATE_RETRY_COUNT

This value is incremented when an attempt to perform an update operation is repeated.

DELETE_RETRY_COUNT

This value is incremented when an attempt to perform a delete operation is repeated.

V$MEM_BTREE_HEADER

This view shows information about a memory BTREE header.

Column name	Type	Description
INDEX_NAME	VARCHAR(128)	The name of the index
INDEX_ID	INTEGER	The index identifier
INDEX_TBS_ID	INTEGER	The tablespace in which the index is stored
TABLE_TBS_ID	INTEGER	The tablespace in which the associated table is stored
IS_UNIQUE	CHAR(1)	Whether an index is a unique key index
IS_NOT_NULL	CHAR(1)	Whether NULL values are allowed
USED_NODE_COUNT	INTEGER	The number of nodes that are being used by an index
PREPARE_NODE_COUNT	INTEGER	The number of nodes that are prepared in advance to meet the demand for nodes
BUILT_TYPE	CHAR(1)	The key type that was used when the index was created

Column Information

INDEX_NAME

This is the name of the index.

INDEX_ID

This is a unique identifier for the index in the system.

INDEX_TBS_ID

This is the identifier of the tablespace in which the index is stored.

TABLE_TBS_ID

This is the identifier of the tablespace containing the table that is related to the index.

IS_UNIQUE

This indicates whether the index is a unique key index. It is set to 'T' to indicate a unique key index, and to 'F' to indicate a duplicate key index.

IS_NOT_NULL

This indicates whether NULL values are allowed. It is set to 'T' for a primary key index, and to 'F' for other kinds of indexes.

USED_NODE_COUNT

This indicates the total number of nodes for the current index. This number increases when a node is split, and decreases when a node is deleted.

PREPARE_NODE_COUNT

This is the number of nodes that are allocated in advance in consideration of system load, based on the number of nodes that have been assigned.

BUILT_TYPE

This indicates whether a key value or a record pointer was used when the index was built. It is set to 'V' to indicate that a key value was used, and to 'P' to indicate that a record pointer was used.

V$MEM_BTREE_NODEPOOL

This view shows information about the node pool for memory BTREE indexes. The node pool manages node allocation and return for all memory BTREE indexes.

Column name	Type	Description
TOTAL_PAGE_COUNT	INTEGER	The total number of pages in the node pool
TOTAL_NODE_COUNT	INTEGER	The total number of nodes in the node pool
FREE_NODE_COUNT	INTEGER	The number of unallocated nodes in the node pool
USED_NODE_COUNT	INTEGER	The number of nodes allocated to indexes

Column name	Type	Description
NODE_SIZE	INTEGER	The size of a node (in bytes)
TOTAL_ALLOC_REQ	BIGINT	The cumulative number of node allocation requests made to the node pool
TOTAL_FREE_REQ	BIGINT	The cumulative number of node deletion requests made to the node pool
FREE_REQ_COUNT	INTEGER	The number of nodes in the node pool waiting to be deleted

Column Information

TOTAL_PAGE_COUNT

This shows the number of pages allocated to the node pool for BTREE indexes.

TOTAL_NODE_COUNT

This indicates the number of nodes allocated to the node pool for BTREE indexes. It is determined by TOTAL_PAGE_COUNT and NODE_SIZE.

FREE_NODE_COUNT

This is the number of nodes that have not been allocated to BTREE indexes, and thus remain in the node pool.

USED_NODE_COUNT

This shows the total number of nodes that are currently allocated to BTREE indexes.

NODE_SIZE

This is the size of a BTREE index node.

TOTAL_ALLOC_REQ

This is the number of node allocation requests that have been made to the node pool. This is the cumulative number since the system was started.

TOTAL_FREE_REQ

This is the number of times the node that was used in the index was deleted and returned to the node pool. This is the accumulated value that is maintained after the system is started.

FREE_REQ_COUNT

This is the number of return requests that have been made to the node pool for nodes that were used for BTREE indexes and then deleted. This is the cumulative number since the system was started

V$MEM_RTREE_HEADER

This view shows information about the header of a memory RTREE index.

Column name	Type	Description
INDEX_NAME	CHAR(40)	The name of the index
INDEX_ID	INTEGER	The index identifier
TABLE_TBS_ID	INTEGER	The identifier of the tablespace in which the table is stored
TREE_MBR_MIN_X	DOUBLE	The minimum X value of the RTREE index
TREE_MBR_MIN_Y	DOUBLE	The minimum Y value of the RTREE index
TREE_MBR_MAX_X	DOUBLE	The maximum X value of the RTREE index
TREE_MBR_MAX_Y	DOUBLE	The maximum Y value of the RTREE index
USED_NODE_COUNT	INTEGER	The number of nodes that are being used by the index
PREPARE_NODE_COUNT	INTEGER	The number of nodes that have been pre-allocated to meet node demand

Column Information

INDEX_NAME

This is the name of the index.

INDEX_ID

This is the identifier of the index. This identifier is unique within the system.

TABLE_TBS_ID

This is the identifier of the tablespace containing the table that is related to the index.

TREE_MBR_MIN_X

This is the minimum X value of the minimum bounding box of the RTREE index.

TREE_MBR_MIN_Y

This is the minimum Y value of the minimum bounding box of the RTREE index.

TREE_MBR_MAX_X

This is the maximum X value of the minimum bounding box of the RTREE index.

TREE_MBR_MAX_Y

This is the maximum Y value of the minimum bounding box of the RTREE index.

USED_NODE_COUNT

This is the total number of nodes being used by the current index. This number increases when a node is split and decreases when a node is deleted.

PREPARE_NODE_COUNT

This is the number of nodes that are allocated in advance in consideration of system load, based on the number of nodes that have been assigned.

V$MEM_RTREE_NODEPOOL

This view shows information about the node pool for memory RTREE indexes. This node pool manages node allocation and return for all memory RTREE indexes.

Column name	Type	Description
TOTAL_PAGE_COUNT	INTEGER	The total number of pages in the node pool
TOTAL_NODE_COUNT	INTEGER	The number of nodes allocated to indexes
FREE_NODE_COUNT	INTEGER	The number of unallocated nodes in the node pool
USED_NODE_COUNT	INTEGER	The number of nodes allocated to indexes
NODE_SIZE	INTEGER	The size of a node (in bytes)
TOTAL_ALLOC_REQ	BIGINT	The cumulative number of node allocation requests made to the node pool
TOTAL_FREE_REQ	BIGINT	The cumulative number of node deletion requests made to the node pool
FREE_REQ_COUNT	INTEGER	The number of nodes in the node pool that are waiting to be deleted

Column Information

TOTAL_PAGE_COUNT

This is the number of pages allocated to the node pool for RTREE indexes.

TOTAL_NODE_COUNT

This is the total number of nodes allocated to the node pool for RTREE indexes. It is determined by TOTAL_PAGE_COUNT and NODE_SIZE.

FREE_NODE_COUNT

This is the number of nodes that have not been allocated to RTREE indexes and thus remain in the node pool.

USED_NODE_COUNT

This is the total number of nodes that are currently allocated to RTREE indexes.

NODE_SIZE

This is the size of an RTREE index node.

TOTAL_ALLOC_REQ

This is the number of node allocation requests that have been made to the node pool. This is the cumulative number since the system was started.

TOTAL_FREE_REQ

This is the number of return requests that have been made to the node pool for nodes that were being used by RTREE indexes and were then deleted. This is the cumulative number since the system was started.

FREE_REQ_COUNT

This is the number of nodes that were being used by RTREE indexes and are waiting to be deleted.

V$MEM_TABLESPACES

This view shows information about tablespaces that exist in memory.

Column name	Type	Description
SPACE_ID	INTEGER	The tablespace identifier
SPACE_NAME	VARCHAR(512)	The name of the tablespace
SPACE_STATUS	INTEGER	The tablespace status
SPACE_SHM_KEY	INTEGER	The share memory key of the tablespace
AUTOEXTEND_MODE	INTEGER	The auto extension mode of the tablespace
AUTOEXTEND_NEXTSIZE	BIGINT	The size (in bytes) by which the tablespace is automatically extended
MAXSIZE	BIGINT	The maximum size of the tablespace (in bytes)
CURRENT_SIZE	BIGINT	The current size of the tablespace (in bytes)
DBFILE_SIZE	DOUBLE	The size of the database image files (in bytes)
DBFILE_COUNT_0	INTEGER	The number of database image files in file group #0

Column name	Type	Description
DBFILE_COUNT_1	INTEGER	The number of database image files in file group #1
TIMESTAMP	VARCHAR(64)	The time point at which the tablespace was created
ALLOC_PAGE_COUNT	BIGINT	The total number of pages in the tablespace
FREE_PAGE_COUNT	BIGINT	The number of free pages in the tablespace
RESTORE_TYPE	BIGINT	How to load the tablespace into memory
CURRENT_DB	INTEGER	A set of files that are the target for ping pong checkpointing
HIGH_LIMIT_PAGE	BIGINT	The maximum number of pages that the tablespace can have
PAGE_COUNT_PER_FILE	BIGINT	The number of pages per database image file
PAGE_COUNT_IN_DIS K	INTEGER	The number of pages that exist on disk

Column Information

SPACE_STATUS

This is a value that indicates the tablespace status. Please refer to V$MEM_TABLESPACE_STATUS_DESC for details.

SPACE_SHM_KEY

This is a shared memory key, which is used when a tablespace is loaded into shared memory.

AUTOEXTEND_MODE

This indicates whether Autoextend mode is enabled. If it is set to 1, Autoextend mode is enabled, whereas if it is set to some other value, Autoextend mode is not enabled..

AUTOEXTEND_NEXTSIZE

When the tablespace is automatically extended, this indicates the size (in bytes) by which the tablespace is automatically extended.

MAXSIZE

This is the maximum size of the tablespace (in bytes).

CURRENT_SIZE

This is the current size of the tablespace (in bytes).

DBFILE_SIZE

This is the size of the database image files for the tablespace (in bytes).

DBFILE_COUNT_0

Because Altibase uses ping pong checkpointing, it maintains two sets of databases image files. This value indicates the number of files in file group #0, which is one of these sets.

DBFILE_COUNT_1

Because Altibase uses ping pong checkpointing, it maintains two sets of databases image files. This value indicates the number of files in file group #1, which is one of these sets.

TIMESTAMP

This timestamp value indicates the time point at which the tablespace was created.

ALLOC_PAGE_COUNT

This is the number of pages in the tablespace.

FREE_PAGE_COUNT

This is the number of free pages in the tablespace.

RESTORE_TYPE

This indicates how the tablespace is loaded into memory. It can have the following values:

Loading Metho	Value	Description
RESTORE_TYPE_DYNAMIC	0	The tablespace is loaded into dynamic memory.
RESTORE_TYPE_SHM_CREATE	1	The shared memory is created and the tablespace is loaded into shared memory.
RESTORE_TYPE_SHM_ATTACH	2	The tablespace is attached to shared memory. Attach shared memory to a process while the database is already in shared memory.

CURRENT_DB

This is the database image file group into which dirty pages (changed pages) are downloaded during checkpointing. It can be 0 or 1.

HIGH_LIMIT_PAGE

This is the maximum number of pages that the tablespace can have.

PAGE_COUNT_PER_FILE

This is the number of pages per database image file.

PAGE_COUNT_IN_DISK

This is the total number of pages in all database image files that exist on disk. Altibase increases the size of a database during checkpointing, rather than directly increasing the size of files on disk. Therefore, the number of database pages that exist in memory can be different from the number of pages on disk.

V$MEM_TABLESPACE_CHECKPOINT_PATHS

This view shows the directory path of the database image files in which changed pages (dirty pages) are recorded during checkpointing for a tablespace.

Column name	Type	Description
SPACE_ID	INTEGER	The tablespace identifier
CHECKPOINT_PATH	VARCHAR(512)	The directory in which the database image files are located

V$MEM_TABLESPACE_STATUS_DESC

This view provides descriptions of values that indicate the memory tablespace status. These are the values that the SPACE_STATUS column in the V$MEM_TABLESPACES view can have.

Column name	Type	Description
STATUS	INTEGER	The status value of memory tablespace
STATUS_DESC	VARCHAR(64)	The description of the status value

Column Information

STATUS

This is the status value of the memory tablespace.

STATUS_DESC

This is a description of the status value of the memory tablespace.

The status values and corresponding descriptions are as follows:

STATUS_DESC	Description
OFFLINE	The tablespace is offline.

STATUS_DESC	Description
ONLINE	The tablespace is online.
DISCARDED	The tablespace has been discarded.
DROPPED	The tablespace has been deleted
BACKUP	The tablespace is being backed up.
CREATING	The tablespace is being created.
DROPPING	A request has been made to delete the tablespace.
DROP_PENDING	The tablespace is being deleted.
SWITCHING_TO_OFFLINE	The tablespace is switching to offline status.
SWITCHING_TO_ONLINE	The tablespace is switching to online status.
BLOCK_BACKUP	The tablespace cannot be backed up. Because another operation is in progress, it is necessary to wait until the other operation is complete before backup can be performed.

V$MUTEX

This view displays statistical information about mutexes, which are related to concurrency control performed by Altibase processes.

Column name	Type	Description
NAME	VARCHAR(64)	The name of the mutex
TRY_COUNT	BIGINT	The number of lock attempts
LOCK_COUNT	BIGINT	The number of successful lock attempts
MISS_COUNT	BIGINT	The number of waits resulting from missed lock attempts
SPIN_VALUE	INTEGER	This field is reserved for future use.
TOTAL_LOCK_TIME_US	BIGINT	The total amount of time this mutex has been locked (in microseconds)
MAX_LOCK_TIME_US	BIGINT	The maximum time elapsed while locking this mutex (in microseconds)
THREAD_ID	VARCHAR(64)	The identifier of a thread currently holding a lock.

V$NLS_PARAMETERS

This view shows NLS (National Language Support)-related information for both the server and client for each session.

Column name	Type	Description
SESSION_ID	BIGINT	The session identifier
NLS_USE	VARCHAR(40)	The client character set
NLS_CHARACTERSET	VARCHAR(40)	The database character set
NLS_NCHAR_CHARACTERSET	VARCHAR(40)	The national character set
NLS_COMP	VARCHAR(7)	How characters are compared
NLS_NCHAR_CONV_EXCP	VARCHAR(7)	How to handle errors that arise when converting character sets
NLS_NCHAR_LITERAL_REPLACE	VARCHAR(7)	Whether to check for the presence of NCHAR literals within SQL statements

Column Information

SESSION_ID

This is the unique number of the session.

NLS_USE

This is the client character set. The default character set should be set when processing character data on the client. The character sets and related NLS_USE settings currently supported by Altibase are as follows:

Language	Character Set	NLS_USE
English (default)	US7ASCII	US7ASCII, ASCII, ENGLISH
Korean	KSC-5601 Complete	KSC5601, KO16KSC5601, KOREAN
	MS Extended Complete	MS949, CP949, WINDOWS949
Japanese	EUC-JP (UNIX)	EUCJP
	Shift-JIS (Windows)	SHIFTJIS
	MS932 (Windows)	MS932, CP932
Chinese	China	GB231280, ZHS16CGB231280, CHINESE, MS936
	Taiwan	BIG5, ZHT16BIG5, TAIWAN
Universal	Unicode (UTF-8)	UTF8, UNICODE

When storing data of a different character set than the database character set, it is important to consider convertibility and compatibility between the individual character sets. Please refer to the *Getting Started* for more detailed information about multilingual support.

NLS_CHARACTERSET

This is the database character set used on the server.

NLS_NCHAR_CHARACTERSET

This is the national character set.

NLS_COMP

This indicates the order in which characters are compared according to how they appear in a dictionary of the language corresponding to the character set that was specified when the database was created. At present, this is useful only when Korean (KSC-5601 Completion or MS Extended Completion) is specified as the database character set.

NLS_NCHAR_CONV_EXCP

This shows how errors are handled when the character set is converted.

NLS_NCHAR_LITERAL_REPLACE

This shows whether the client will check whether NCHAR literals exist within a SQL statement. If this is TRUE, the client always checks whether NCHAR literals exist, and convert the remainder of the SQL statement, other than the NCHAR literals, to the database character set. If this is FALSE, the client doesn't check this, and convert the entire SQL statement to to the database character set.

V$NLS_TERRITORY

This view stores the names of territories available to be set for the database or the current session.

Column name	Type	Description
NAME	VARCHAR(40)	The name of the territory available to be set

V$OBSOLETE_BACKUP_INFO

This view displays information about backups which are no longer required to be retained.

Since columns of this view are part of the V$BACKUP_INFO performance view, please refer to column information about the V$BACKUP_INFO performance view for more information.

Column name	Type	Description
BEGIN_BACKUP_TIME	CHAR(24)	The start time of backup
END_BACKUP_TIME	CHAR(24)	The completion time of backup
INCREMENTAL_BACKUP_CHUNK_COUNT	INTEGER	The incremental chunk size

Column name	Type	Description
BACKUP_TARGET	INTEGER	The backup target
BACKUP_LEVEL	INTEGER	The backup level
BACKUP_TYPE	INTEGER	The backup type
TABLESPACE_ID	INTEGER	The backup target tablespace ID
FILE_ID	INTEGER	The backup target datafile ID
BACKUP_TAG	CHAR(128)	The backup tag name
BACKUP_FILE	CHAR(512)	The backup file

V$PKGTEXT

This view contains information about strings of the packages executed on the system.

Column name	Type	Description
PACKAGE_OID	BIGINT	The package identifier
PIECE	INTEGER	The serial number of the string piece
TEXT	VARCHAR(64)	The string piece of the package statement

Column Information

PACKAGE_OID

This is the object identifier which only points to the package; the OID.

PIECE

The entire statement of the package is split into strings with the length of 64 bytes and saved. PIECE is the serial number of the 64-byte split pieces, and starts from 0.

TEXT

This is the column which displays the text of the 64-byte text pieces which are parts of the package text.

V$PLANTEXT

This view displays information about execution plans for SQL statements that are executed by the server.

Column name	Type	Description
SID	INTEGER	The session identifier

Column name	Type	Description
STMT_ID	INTEGER	The statement identifier
PIECE	INTEGER	The serial number for the fragment of execution plan text
TEXT	VARCHAR(64)	A fragment of execution plan text

Column Information

SID

This is the identifier of the session.

STMT_ID

This is the identifier of the statement.

PIECE

A complete execution plan for one statement is divided into text fragments 64 bytes long and then saved. PIECE shows the serial numbers for these 64-byte fragments, starting from 0.

TEXT

This shows the contents of the 64-byte text fragment that is part of the execution plan statement.

V$PROCTEXT

This view displays information about stored procedures being used by the system.

Column name	Type	Description
PROC_OID	BIGINT	The object identifier of a stored procedure
PIECE	INTEGER	The serial number for the stored procedure fragment
TEXT	VARCHAR(64)	A fragment of the stored procedure text

Column Information

PROC_OID

The identifier of a stored procedure or stored function, which is the same as a PROC_OID value in the SYS_PROCEDURES_ meta table.

MODIFY_COUNT

Incremented by 1 each time a stored procedure or function is recreated or recompiled. The initial value is 0.

STATUS

The value indicating whether a stored procedure or function can be executed. VALID indicates that it is executable. Refer to the description of the STATUS column in the SYS_PROCEDURES_ meta table.

SESSION_ID

The ID of the session that changed the status of the stored procedure or funciton to INVALID. If the status has never changed, this value is 0 or -1.

PROC_TYPE

The type of stored procedure. The possible values are:

- NORMAL : Normal procedure
- EXTERNAL C : C/C++ External Procedure
- INTERNAL C : C/C++ Internal Procedure
- UNKNOWN : If the compilation of the stored procedure fails when starting the server, the internal procedure type is not known, so it is marked UNKNOWN. Subsequently, when compiled and in VALID status, the correct type is set.

V$PROCINFO

Column name	Type	Description
PROC_OID	BIGINT	The object identifier of the stored procedure
MODIFY_COUNT	INTEGER	The number of times a stored procedure was recreated or recompiled
STATUS	VARCHAR(7)	The status of the object. If INVALID, it is not executable
SESSION_ID	INTEGER	The ID of the session that changed the STATUS of the stored procedure
PROC_TYPE	VARCHAR(10)	The type of stored procedure

Column Information

PROC_OID

This is an OID, which is a unique object identifier for a stored procedure.

PIECE

The complete text for a stored procedure is divided into text fragments 64 bytes long and then saved. PIECE shows the serial numbers for these 64-byte fragments, starting from 0.

TEXT

This shows the contents of the 64-byte text fragment that is part of the stored procedure text.

V$PROPERTY

This view displays information about all internally set Altibase properties.

Column name	Type	Description
NAME	VARCHAR(256)	The property name
STOREDCOUNT	INTEGER	The number of values set for the property
ATTR	BIGINT	The property attribute
MIN	VARCHAR(256)	The minimum value
MAX	VARCHAR(256)	The maximum value
VALUE1	VARCHAR(256)	The first value
VALUE2	VARCHAR(256)	The second value
VALUE3	VARCHAR(256)	The third value
VALUE4	VARCHAR(256)	The fourth value
VALUE5	VARCHAR(256)	The fifth value
VALUE6	VARCHAR(256)	The sixth value
VALUE7	VARCHAR(256)	The seventh value
VALUE8	VARCHAR(256)	The eighth value

Column Information

NAME

This is the name of the property.

STOREDCOUNT

STOREDCOUNT displays the number of values set in the property. A property can have up to eight duplicate values.

ATTR

This is the attribute of the property.

MIN

This is the minimum value that the property can have.

MAX

This is the maximum value that the property can have.

VALUE1 ~ 8

The actual values set for the property.

V$REPEXEC

This view displays information related to the replication manager.

Column name	Type	Description
PORT	INTEGER	The port number currently being used
MAX_SENDER_COUNT	INTEGER	The maximum number of Sender threads
MAX_RECEIVER_COUNT	INTEGER	The maximum number of Receiver threads

Column Information

PORT

The number of the port through which the replication manager on the local server receives replication requests from the remote server.

MAX_SENDER_COUNT

This is the maximum number of replication Sender threads that can be created on the local server.

MAX_RECEIVER_COUNT

This is the maximum number of replication Receiver threads that can be created on the local server.

V$REPGAP

This shows the difference between the most recently created log record and the log record currently being processed by the replication Sender. Please note that this information is only available while the replication Sender thread is active.

Column name	Type	Description
REP_NAME	VARCHAR(40)	The name of the replication object
START_FLAG	BIGINT	Startup options
REP_LAST_SN	BIGINT	The sequence number of the last log record

Column name	Type	Description
REP_SN	BIGINT	The sequence number of the log record currently being sent
REP_GAP	BIGINT	The actual size of the log file corresponding to the replication gap (Unit: Unit set in property REPLICATION_GAP_UNIT)
REP_GAP_SIZE	BIGINT	The actual size of the log file corresponding to the replication gap (bytes)
READ_LFG_ID	INTEGER	The log file group currently being read (Not used, 0)
READ_FILE_NO	INTEGER	The log file number currently being read
READ_OFFSET	INTEGER	The location currently being read

Column Information

REP_NAME

This is the name of the replication object on the local server.

START_FLAG

This is a replication startup option for use when replication is started on the local server. The following values are possible:

- NORMAL: 0
- QUICK: 1
- SYNC: 2
- SYNC_ONLY: 3
- SYNC RUN: 4
- SYNC END: 5
- RECOVERY from Replication: 6
- OFFLINE: 7
- PARALLEL: 8

REP_LAST_SN

This is the sequence number of the log record that was most recently written in response to a transaction on the local server.

REP_SN

This is the sequence number of the log record that is currently being sent by the replication Sender on the local server.

REP_GAP

This shows the interval between the log sequence numbers of REP_LAST_SN and REP_SN. In other words, this is the interval between the log record that was most recently written due to a transaction on the local server and the log record that is currently being sent by the replication Sender thread.

REP_GAP_SIZE

This shows the log file size of the replication gap, in bytes.

READ_FILE_NO

This is the log file number which is currently being read by the replication sender. However, this is not updated while the replication sender is reading the replication log in the buffer. To check to see if the log is being read in the replication log buffer, verify the READ_SN value is between BUFFER_MIN_SN and BUFFER_MAX_SN.

READ_OFFSET

This indicates the location in the log file that is currently being read.

V$REPGAP_PARALLEL

This view shows the difference between the most recently created log record and the log record currently being processed by replication Sender threads working in parallel. Please note that this information is only available when multiple replication Sender threads are working in parallel.

Column name	Type	Description
REP_NAME	VARCHAR(40)	The name of the replication
CURRENT_TYPE	VARCHAR(9)	The type of the replication Sender thread
REP_LAST_SN	BIGINT	The last log file number
REP_SN	BIGINT	The sequence number of the log record currently being sent
REP_GAP	BIGINT	The actual size of the log file corresponding to the replication gap (Unit: unit set in property REPLCIATION_GAP_UNIT)
REP_GAP_SIZE	BIGINT	The actual size of the log file corresponding to the replication gap (bytes)
READ_LFG_ID	INTEGER	The identifier of the log file group currently being read
READ_FILE_NO	INTEGER	The log file number currently being read

Column name	Type	Description
READ_OFFSET	INTEGER	The current reading offset
PARALLEL_ID	INTEGER	The identifier of one of several threads working in parallel for one Sender

Column Information

REP_NAME

This is the name of the replication object on the local server.

CURRENT_TYPE

This can have one of the following values, which denote the current status of the replication Sender thread.

- NORMAL: This means that the Sender thread analyzes transaction logs and converts them to XLOGs on the active server. The Sender thread then transfers the XLOGs to a standby server.

- QUICK: This value can be returned when replication was started with the QUICKSTART option, and indicates the state in which the starting location is being changed so that the Sender thread will ignore old logs and start sending from the most recent log. After the starting location is changed, NORMAL will be returned, rather than this value.

- SYNC: This value can be returned when replication was started with the SYNC option. After synchronization is complete, NORMAL (LAZY mode) or PARALLEL (EAGER mode) will be returned, rather than this value.

- SYNC_ONLY: This value can be returned when replication was started with the SYNC ONLY option. After synchronization is complete, the Sender thread will be terminated.

- RECOVERY: This value indicates that the Sender thread is running in order to restore data that were lost on another server.

- OFFLINE: This value indicates that the Sender thread is running in order to read logs on the active server when the active server is offline and apply them to the standby server.

- PARALLEL: This value indicates that several Sender threads are sending XLOGs pertaining to the table(s) that is (or are) being replicated in parallel. This value can be returned when replication was started in EAGER mode with the PARALLEL option. It is different from the PARALLEL option which can be specified when starting replication with the SYNC or SYNC_ONLY option.

REP_LAST_SN

This is the most recent log record sequence number on the local server.

REP_SN

This is the sequence number of the log record that is currently being sent by the replication Sender on the local server.

REP_GAP

This is the difference between the log serial number returned by REP_LAST_SN and that returned by REP_SN. In other words, this is the gap between the log record that was most recently written by a transaction on the local server and the log record that is currently being sent by the replication Sender thread.

REP_GAP_SIZE

This is the log file size of the replication gap, in bytes.

READ_FILE_NO

This indicates the number of the log file that is currently being read.

READ_OFFSET

This indicates the current location in the log file that is currently being read.

PARALLEL_ID

This is the identifier of one of several threads working in parallel for one Sender.

V$REPLOGBUFFER

This view displays information about the state of the log buffer used by the replication Sender while the replication Sender thread is working

Column name	Type	Description
REP_NAME	VARCHAR(40)	The name of the replication object
BUFFER_MIN_SN	BIGINT	The lowest log sequence number in the buffer that is being used by the replication Sender
READ_SN	BIGINT	The sequence number of the log record to be read next by the replication Sender thread
BUFFER_MAX_SN	BIGINT	The highest log sequence number in the buffer that is being used by the replication Sender

Column Information

REP_NAME

This is the name of the replication object on the local server.

BUFFER_MIN_SN

This is the lowest of the sequence numbers of log records saved in the log buffer that is used for replication.

READ_SN

This is the sequence number of the log record that is to be read next by the replication Sender thread in the log buffer that is being used for replication.

BUFFRT_MAX_SN

This is the highest of the sequence numbers of log records saved in the log buffer that is being used for replication.

V$REPOFFLINE_STATUS

This view shows the status of offline replication.

Column name	Type	Description
REP_NAME	VARCHAR(40)	The name of the replication object
STATUS	BIGINT	The status of offline replication execution
SUCCESS_TIME	INTEGER	The time taken for offline replication to execute successfully

Column Information

REP_NAME

This is the name of the replication object on the local server.

STATUS

This is the status of offline replication.

- 0: offline replication has not been started
- 1: offline replication has been started
- 2: offline replication has completed
- 3: offline replication failed

SUCCESS_TIME

This is the time point at which the most recent successful execution of offline replication occurred. It is based on the system time. In the case where replication was successfully started and completed, it is the time taken for replication to complete, and is 0 otherwise.

V$REPRECEIVER

This view displays information about the replication Receiver.

Column name	Type	Description
REP_NAME	VARCHAR(40)	The name of the replication object
MY_IP	VARCHAR(64)	The IP address of the local sever

Column name	Type	Description
MY_PORT	INTEGER	The port number on the local server
PEER_IP	VARCHAR(64)	The IP address on the remote server
PEER_PORT	INTEGER	The port number on the remote server
APPLY_XSN	BIGINT	The XSN currently being processed
INSERT_SUCCESS_COUNT	BIGINT	The number of INSERT log records successfully applied to the local database by the replication Receiver thread
INSERT_FAILURE_COUNT	BIGINT	The number of INSERT log records that could not be applied to the local database by the replication Receiver thread
UPDATE _SUCCESS_COUNT	BIGINT	The number of UPDATE log records successfully applied to the local database by the replication Receiver thread
UPDATE_FAILURE_COUNT	BIGINT	The number of UPDATE log records that could not be applied to the local database by the replication Receiver thread
DELETE_SUCCESS_COUNT	BIGINT	The number of DELETE log records successfully applied to the local database by the replication Receiver thread
DELETE_FAILURE_COUNT	BIGINT	The number of DELETE log records that could not be applied to the local database by the replication Receiver thread
PARALLEL_ID	INTEGER	Always displays 0
SQL_APPLY_TABLE_COUNT	INTEGER	The number of tables operating in SQQL reflection mode
APPLIER_INIT_BUFFER_USAGE	BIGINT	The current size of the queue waiting on the parallel applicator (unit: byte)

Column Information

REP_NAME

This is the name of the replication object on the local server.

MY_IP

This is the IP address of the local server.

MY_PORT

This is the port number being used by the Receiver thread on the local server.

PEER_IP

This is the IP address of the remote server.

PEER_PORT

This is the port number being used by the Sender thread on the remote server.

APPLY_XSN

This shows the XLog sequence number (XSN) of the XLog that was sent by the Sender thread on the remote server and is being used by the Receiver thread on the local server.

INSERT_SUCCESS_COUNT

This is the number of INSERT log records that were successfully applied to the local database by the replication Receiver thread.

This number is not dependent on whether statements are committed or rolled back. In other words, if a statement is rolled back, this number is not decreased.

INSERT_FAILURE_COUNT

This is the number of INSERT log records (including conflicts) that could not be applied to the local database by the replication Receiver thread.

This number is not dependent on whether statements are committed or rolled back. In other words, if a statement is rolled back, this number is not decreased.

UPDATE_SUCCESS_COUNT

This is the number of UPDATE log records that were successfully applied to the local database by the replication Receiver thread.

This number is not dependent on whether statements are committed or rolled back. In other words, if a statement is rolled back, this number is not decreased.

UPDATE_FAILURE_COUNT

This is the number of UPDATE log records (including conflicts) that could not be applied to the local database by the replication Receiver thread.

This number is not dependent on whether statements are committed or rolled back. In other words, if a statement is rolled back, this number is not decreased.

DELETE_SUCCESS_COUNT

This is the number of DELETE log records that were successfully applied to the local database by the replication Receiver thread.

This number is not dependent on whether statements are committed or rolled back. In other words, if a statement is rolled back, this number is not decreased.

DELETE_FAILURE_COUNT

This is the number of DELETE log records that were successfully applied to the local database by the replication Receiver thread.

This number is not dependent on whether statements are committed or rolled back. In other words, if a statement is rolled back, this number is not decreased.

PARALLE_ID

Always displays 0.

In eager mode, this is the same Receiver as the replication Receiver whose PARALLEL_ID is 0 in V$REPRECEIVER_PARALLEL. For other modes, this value is meaningless.

SQL_APPLY_TABLE_COUNT

This is the number of tables running in SQL refleciton mode.

APPLIER_INIT_BUFFER_USAGE

This is the total memory usage of the XLog allocated to the applier thread, when using replication with the parallel applier option (in bytes)

V$REPRECEIVER_COLUMN

This view shows information about columns that are replication targets used by the replication Receiver.

Column name	Type	Description
REP_NAME	VARCHAR(40)	The name of the replication
USER_NAME	VARCHAR(128)	The user name
TABLE_NAME	VARCHAR(128)	The table nam
PARTITION_NAME	VARCHAR(128)	The name of the partition
COLUMN_NAME	VARCHAR(128)	The column name
APPLY_MODE	INTEGER	0: Binary mode 1: SQL mode

Column Information

REP_NAME

This is the name of the replication object on the local server.

USER_NAME

This is the user name of the owner of the table that is the target of replication on the local server. Its value corresponds to a USER_NAME in the SYS_USERS_ meta table.

TABLE_NAME

This is the name of a table that is the target of replication on the local server. It corresponds to a TABLE_NAME in the SYS_TABLES_ meta table.

PARTITION_NAME

This is the name of the partition that is the target for replication on the local server.

COLUMN_NAME

This is the name of the column that is the target of replication on the local server.

APPLY_MODE

This mode reflects data in a table.

- 0: Binary mode
- 1: SQL mode

V$REPRECEIVER_PARALLEL

This view displays information about replication Receiver threads working in parallel.

Column name	Type	Description
REP_NAME	VARCHAR(40)	The name of the replication object
MY_IP	VARCHAR(64)	The IP address of the local server
MY_PORT	INTEGER	The port number on the local server
PEER_IP	VARCHAR(64)	The IP address of the remote server
PEER_PORT	INTEGER	The port number on the remote server
APPLY_XSN	BIGINT	The XSN currently being processed
INSERT_SUCCESS_COUNT	BIGINT	The number of INSERT transactions successfully applied to the local database by the replication Receiver thread.

Column name	Type	Description
INSERT_FAILURE_COUNT	BIGINT	The number of INSERT transactions that could not be applied to the local database by the replication Receiver thread.
UPDATE _SUCCESS_COUNT	BIGINT	The number of UPDATE transactions successfully applied to the local database by the replication Receiver thread.
UPDATE_FAILURE_COUNT	BIGINT	The number of UPDATE transactions that could not be applied to the local database by the replication Receiver thread.
DELETE_SUCCESS_COUNT	BIGINT	The number of DELETE transactions successfully applied to the local database by the replication Receiver thread.
DELETE_FAILURE_COUNT	BIGINT	The number of DELETE transactions that could not be applied to the local database by the replication Receiver thread.
PARALLEL_ID	INTEGER	The identifier of one of several replication Receiver threads working in parallel

Column Information

REP_NAME

This is the name of the replication object.

MY_IP

This is the IP address of the local server.

MY_PORT

This is the port number used by the Receiver on the local server.

PEER_IP

This is the IP address of the remote server.

PEER_PORT

This is the port number used by the Sender on the remote server.

APPLY_XSN

This shows the XLog sequence number of the XLog that was sent by a Sender thread on the remote server and is being applied by the Receiver thread on the local server.

INSERT_SUCCESS_COUNT

This is the number of INSERT transactions that were successfully applied to the local database by the replication Receiver thread.

This number is not dependent on whether statements are committed or rolled back. In other words, if a statement is rolled back, this number is not decreased.

INSERT_FAILURE_COUNT

This is the number of INSERT transactions (including conflicts) that could not be applied to the local database by the replication Receiver thread.

This number is not dependent on whether statements are committed or rolled back. In other words, if a statement is rolled back, this number is not decreased.

UPDATE_SUCCESS_COUNT

This is the number of UPDATE transactions that were successfully applied to the local database by the replication Receiver thread.

This number is not dependent on whether statements are committed or rolled back. In other words, if a statement is rolled back, this number is not decreased.

UPDATE_FAILURE_COUNT

This is the number of INSERT transactions (including conflicts) that could not be applied to the local database by the replication Receiver thread.

This number is not dependent on whether statements are committed or rolled back. In other words, if a statement is rolled back, this number is not decreased.

DELETE_SUCCESS_COUNT

This is the number of DELETE transactions that were successfully applied to the local database by the replication Receiver thread.

This number is not dependent on whether statements are committed or rolled back. In other words, if a statement is rolled back, this number is not decreased.

DELETE_FAILURE_COUNT

This is the number of INSERT transactions (including conflicts) that could not be applied to the local database by the replication Receiver thread.

This number is not dependent on whether statements are committed or rolled back. In other words, if a statement is rolled back, this number is not decreased.

PARALLEL_ID

This is the identifier of one of several replication Receivers having the same name.

V$REPRECEIVER_PARALLEL_APPLY

This view displays information about replication applier threads.

Column name	Type	Description
REP_NAME	VARCHAR(40)	The name of the replication object
PARALLEL_APPLIER_INDEX	INTEGER	The applier number
APPLY_XSN	BIGINT	The XSN currently being processed
INSERT_SUCCESS_COUNT	BIGINT	The number of INSERT transactions successfully applied to the local database by the replication Receiver thread.
INSERT_FAILURE_COUNT	BIGINT	The number of INSERT transactions that could not be applied to the local database by the replication Receiver thread.
UPDATE_SUCCESS_COUNT	BIGINT	The number of UPDATE transactions successfully applied to the local database by the replication Receiver thread.
UPDATE_FAILURE_COUNT	BIGINT	The number of UPDATE transactions that could not be applied to the local database by the replication Receiver thread.
DELETE_SUCCESS_COUNT	BIGINT	The number of DELETE transactions successfully applied to the local database by the replication Receiver thread.
DELETE_FAILURE_COUNT	BIGINT	The number of DELETE transactions that could not be applied to the local database by the replication Receiver thread.

Column Information

For more detailed information, please refer to V$REPRECEIVER.

V$REPRECEIVER_STATISTICS

This view shows statistical information about the time that it takes for replication Receivers to perform various tasks. When the TIMED_STATISTICS property is set to 1, cumulative statistics are maintained in this view. The interval at which this statistical information is updated is determined by the TIMER_THREAD_RESOLUTION and TIMER_RUNNING_LEVEL properties.

Column name	Type	Description
REP_NAME	VARCHAR(40)	This is the name of the replication object.

Column name	Type	Description
PARALLEL_ID	INTEGER	This is the identifier of one of several replication Receiver threads working in parallel.
RECV_XLOG	BIGINT	This is the cumulative amount of time taken to receive XLogs.
CONVERT_ENDIAN	BIGINT	This is the cumulative amount of time taken to perform byte order conversion.
BEGIN_TRANSACTION	BIGINT	This is the cumulative amount of time taken to begin transactions.
COMMIT_TRANSACTION	BIGINT	This is the cumulative amount of time taken to commit transactions.
ABORT_TRANSACTION	BIGINT	This is the cumulative amount of time taken to roll back transactions.
OPEN_TABLE_CURSOR	BIGINT	This is the cumulative amount of time taken to open table cursors.
CLOSE_TABLE_CURSOR	BIGINT	This is the cumulative amount of time taken to close table cursors.
INSERT_ROW	BIGINT	This is the cumulative amount of time taken to replay logs for INSERT statements.
UPDATE_ROW	BIGINT	This is the cumulative amount of time taken to replay logs for UPDATE statements.
DELETE_ROW	BIGINT	This is the cumulative amount of time taken to replay logs for DELETE statements.
OPEN_LOB_CURSOR	BIGINT	This is the cumulative amount of time taken to open LOB cursors.
PREPARE_LOB_WRITING	BIGINT	This is the cumulative amount of time taken to prepare to write LOBs.
WRITE_LOB_PIECE	BIGINT	This is the cumulative amount of time taken to write LOB pieces.
FINISH_LOB_WRITE	BIGINT	This is the cumulative amount of time taken to finish writing LOBs.
CLOSE_LOB_CURSOR	BIGINT	This is the cumulative amount of time taken to close LOB cursors.
COMPARE_IMAGE	BIGINT	This is the cumulative amount of time taken to compare data in order to resolve conflicts.

Column name	Type	Description
SEND_ACK	BIGINT	This is the cumulative amount of time taken to send ACK.

Column Information

REP_NAME

This is the name of the replication object.

PARALLEL_ID

This is the identifier of one of several replication Receiver threads having the same replication name. When parallel Receiver threads are working in eager mode, a unique ID is given to each thread.

RECV_XLOG

This is the cumulative amount of time taken to receive XLogs from Sender Thread(s). This value includes the amount of time spent waiting for new XLogs to arrive at Receiver Thread(s).

CONVERT_ENDIAN

This is the cumulative amount of time taken to perform byte order conversions. Byte order conversion is performed when the byte order of the platform on which the Sender is running is different from that of the Receiver.

BEGIN_TRANSACTION

This is the cumulative amount of time taken to begin transactions.

COMMIT_TRANSACTION

This is the cumulative amount of time taken to commit transactions.

ABORT_TRANSACTION

This is the cumulative amount of time taken to roll back transactions.

OPEN_TABLE_CURSOR

This is the cumulative amount of time taken to open table cursors.

CLOSE_TABLE_CURSOR

This is the cumulative amount of time taken to close table cursors.

INSERT_ROW

This is the cumulative amount of time that Receiver thread(s) have taken to replay logs for INSERT statements.

UPDATE_ROW

This is the cumulative amount of time that Receiver thread(s) have taken to replay logs for UPDATE statements.

DELETE_ROW

This is the cumulative amount of time that Receiver thread(s) have taken to replay logs for DELETE statements.

OPEN_LOB_CURSOR

This is the cumulative amount of time taken to open LOB cursors.

PREPARE_LOB_WRITING

This is the cumulative amount of time taken to prepare to write LOBs.

WRITE_LOB_PIECE

This is the cumulative amount of time taken to write LOB pieces.

FINISH_LOB_WRITE

This is the cumulative amount of time taken to finish writing LOBs.

CLOSE_LOB_CURSOR

This is the cumulative amount of time taken to close LOB cursors.

COMPARE_IMAGE

This is the cumulative amount of time taken to compare data in order to resolve data conflicts.

SEND_ACK

This is the cumulative amount of time taken to send ACK to Sender Thread(s).

V$REPRECEIVER_TRANSTBL

This view displays information about the replication Receiver's transaction table.

Column name	Type	Description
REP_NAME	VARCHAR(40)	The name of the replication object
LOCAL_TID	BIGINT	The local transaction identifier
REMOTE_TID	BIGINT	The remote transaction identifier
BEGIN_FLAG	INTEGER	Not currently used
BEGIN_SN	BIGINT	The first log record sequence number of the transaction

Column name	Type	Description
PARALLEL_ID	INTEGER	The identifier of one of multiple replication receiver threads operating in parallel among the identical replication objects
PARALLEL_APPLIER_INDEX	INTEGER	The number of the applier that is running the transaction

Column Information

REP_NAME

This is the name of the replication object on the local server.

LOCAL_TID

This is the identifier of the transaction that is being executed on the local server.

REMOTE_TID

This is the identifier of the transaction that is executed on the remote server. It may or may not have already finished being executed.

V$REPRECEIVER_TRANSTBL_PARALLEL

This view displays information about transaction tables used by multiple replication Receiver threads working in parallel.

Column name	Type	Description
REP_NAME	VARCHAR(40)	The name of the replication object
LOCAL_TID	INTEGER	The local transaction identifier
REMOTE_TID	INTEGER	The remote transaction identifier
BEGIN_FLAG	INTEGER	Not currently used
BEGIN_SN	BIGINT	The first log record sequence number of the transaction
PARALLEL_ID	INTEGER	The identifier of one of several Receivers having the same name

Column Information

REP_NAME

This is the name of the replication object on the local server.

LOCAL_TID

This is the identifier of a transaction that is being executed on the local server.

REMOTE_TID

This is the identifier of a transaction that is executed on the remote server. It may or may not have already finished being executed.

PARALLEL_ID

This is the identifier of one of several replication Receivers working in parallel.

V$REPRECOVERY

This view shows information pertaining to recovery using replication.

Column name	Type	Description
REP_NAME	VARCHAR(40)	The name of the replication object
STATUS	INTEGER	The present status of recovery 1: Generating recovery information 2: Recovery request pending 3: Recovery in progress
START_XSN	BIGINT	The first SN sent for recovery
XSN	BIGINT	The SN currently being sent for recovery
END_XSN	BIGINT	The last SN sent for recovery
RECOVERY_SENDER_IP	VARCHAR(64)	The IP address of the Sender for recovery of the local server
PEER_IP	VARCHAR(64)	The IP address of the Receiver for recovery of the remote server
RECOVERY_SENDER_PORT	INTEGER	The port number used by the Sender for recovery of the local server
PEER_PORT	INTEGER	The port number used by the Receiver for recovery of the remote server

Column Information

REP_NAME

This is the name of the replication object on the local server.

STATUS

This is the present status of replication Sender threads on the local server.

- 1: Recovery information is being generated
- 2: A recovery request is waiting
- 3: Recovery is underway

START_XSN

This shows the sequence number of the first log record to be sent by the Sender thread for recovery of the local server.

XSN

This shows the sequence number of the log record currently being sent by the Sender thread for recovery of the local server.

END_XSN

This shows the sequence number of the last log record to be sent by the Sender thread for recovery of the local server.

RECOVERY_SENDER_IP

This is the IP address of the Sender for recovery of the local server.

PEER_IP

This is the IP address of the remote server for recovery of the remote server.

RECOVERY_SENDER_PORT

This is the port number being used by the Sender thread for recovery of the local server.

PEER_PORT

This is the port number being used by the Receiver thread for recovery of the remote server.

V$REPSENDER

This view displays information about the replication Sender.

Column name	Type	Description
REP_NAME	VARCHAR(40)	The name of the replication object
START_FLAG	BIGINT	A flag indicating startup options
NET_ERROR_FLAG	BIGINT	A flag indicating a network error
XSN	BIGINT	The sequence number of the log record being sent

Column name	Type	Description
COMMIT_XSN	BIGINT	The sequence number of the committed log record that was most recently read by the Sender
STATUS	BIGINT	The current status of the replication Sender
SENDER_IP	VARCHAR(64)	The IP address of the Sender
PEER_IP	VARCHAR(64)	The IP address of the remote server
SENDER_PORT	INTEGER	The port number used by the Sender
PEER_PORT	INTEGER	The port number used by the Receiver on the remote server
READ_LOG_COUNT	BIGINT	The number of logs that have been read
SEND_LOG_COUNT	BIGINT	The number of logs that have been read and sent
REPL_MODE	VARCHAR(7)	The replication mode specified by the user
ACT_REPL_MODE	VARCHAR(7)	The actual replication mode

Column Information

REP_NAME

This is the name of the replication object on the local server.

START_FLAG

This is a flag indicating the replication startup options on the local server. It can have the following values:

- : 0
- QUICK: 1
- SYNC: 2
- SYNC_ONLY: 3
- SYNC RUN : 4
- SYNC END : 5
- RECOVERY from Replication : 6
- OFFLINE: 7
- PARALLEL: 8

NET_ERROR_FLAG

This indicates whether a network error has occurred. The default value is 0; 1 indicates that an error has occurred.

XSN

This is the sequence number of the log record that is currently being sent by the replication Sender thread on the local server.

COMMIT_XSN

This is the sequence number of the committed log record that was most recently read by the replication Sender.

STATUS

This is the current status of the replication Sender on the local server. It can have the following values:

- 0: STOP
- 1: RUN
- 2: RETRY
- 3: FAILBACK NORMAL
- 4: FAILBACK MASTER
- 5: FAILBACK SLAVE
- 6: SYNC
- 7: FAILBACK EAGER
- 8: FAILBACK FLUSH
- 9: IDLE

SENDER_IP

This is the IP address of the local server.

PEER_IP

This is the IP address of the remote server.

SENDER_PORT

This is the port number used by the replication Sender thread on the local server.

PEER_PORT

This is the port number used by the replication Receiver thread on the remote server.

READ_LOG_COUNT

This is the number of log records that have been read by the Sender thread on the local server.

SEND_LOG_COUNT

This is the number of log records that have been read and sent by the Sender thread on the local server.

REPL_MODE

This indicates the replication mode set by the user. The type of replicaiton mode is LAZY or EAGER.

For more detailed information about the replication mode, please refer to the *Replication Manual*.

ACT_REPL_MODE

This represents the replication mode in operation and may be different from REPL_MODE.

When the replication mdoe is set to EAGER, if there is a replication gap due to a failure, replication will be operate in LAZY mode.

Otherwise, it is the same as the value of REPL_MODE.

V$REPSENDER_PARALLEL

This view displays information about replication Sender threads working in parallel.

Column name	Type	Description
REP_NAME	VARCHAR(40)	The name of the replication object
CURRENT_TYPE	VARCHAR(9)	The type of the replication Sender thread
NET_ERROR_FLAG	BIGINT	A flag indicating a network error
XSN	BIGINT	The sequence number of the log record currently being sent
COMMIT_XSN	BIGINT	The sequence number of the most recently committed log record
STATUS	VARCHAR(15)	The current status of the replication Sender
SENDER_IP	VARCHAR(64)	The IP address of the Sender
PEER_IP	VARCHAR(64)	The IP address of the remote server
SENDER_PORT	INTEGER	The port number used by the Sender
PEER_PORT	INTEGER	The port number used by the Receiver on the remote server
READ_LOG_COUNT	BIGINT	The number of logs that have been read
SEND_LOG_COUNT	BIGINT	The number of logs that have been read and transmitted
REPL_MODE	VARCHAR(7)	The current replication mode

Column name	Type	Description
PARALLEL_ID	INTEGER	The identifier of one of several replication Senders having the same name

Column Information

REP_NAME

This is the name of the replication object on the local server.

CURRENT_TYPE

Please refer to the description of the CURRENT_TYPE column in the V$REPGAP_PARALLEL performance view.

NET_ERROR_FLAG

This indicates whether a network error has occurred. The default value is 0; 1 indicates that an error has occurred.

XSN

This is the sequence number of the log record that is currently being sent by the corresponding replication Sender thread on the local server.

COMMIT_XSN

This is the sequence number of the committed log record that was most recently read by this Sender thread.

STATUS

This is the current status of the replication Sender on the local server. It can have the following values:

- 0: STOP
- 1: RUN
- 2: RETRY
- 3: FAILBACK NORMAL
- 4: FAILBACK MASTER
- 5: FAILBACK SLAVE
- 6: SYNC
- 7: FAILBACK EAGER
- 8: FAILBACK FLUSH
- 9: IDLE

SENDER_IP

This is the IP address of the local server.

PEER_IP

This is the IP address of the remote server.

SENDER_PORT

This is the port number used by this replication Sender thread on the local server.

PEER_PORT

This is the port number used by the corresponding replication Receiver thread on the remote server.

READ_LOG_COUNT

This is the number of log records read by this Sender thread on the local server.

SEND_LOG_COUNT

This is the number of log records read and transmitted by this Sender thread on the local server.

REPL_MODE

This is the replication mode. It can be set to LAZY or EAGER.

For more detailed information about replication modes, please refer to the *Replication Manual*.

PARALLEL_ID

This is the identifier of one of several replication Senders working in parallel.

V$REPSENDER_SENT_LOG_COUNT

This performance view displays the number of logs sent by the replication Sender, sorted by the DML type. Whenever the number of replication logs specified for the REPLICATION_SENDER_LOG_COUNT_PERIOD property is sent, the data of this performance view is updated.

For parallel replication in EAGER mode, only the information about the Parent Sender is displayed in this performance view; information about each Sender thread is displayed in the V$REPSENDER_SENT_LOG_COUNT_PARALLEL performance view.

Column name	Type	Description
REP_NAME	VARCHAR(40)	This is the name of the replication object.
CURRENT_TYPE	VARCHAR(9)	This is the type of the replication Sender thread.
TABLE_OID	BIGINT	This is the table object identifier.
INSERT_LOG_COUNT	INTEGER	This is the table object identifier.
DELETE_LOG_COUNT	INTEGER	This is the number of DELETE logs sent.

Column name	Type	Description
UPDATE_LOG_COUNT	INTEGER	This is the number of UPDATE logs sent.
LOB_LOG_COUNT	INTEGER	This is the number of LOB-related logs sent.

Column Information

REP_NAME

This is the name of the replication object created on the local server.

CURRENT_TYPE

Please refer to the description of the CURRENT_TYPE column in the V$REPGAP_PARALLEL performance view.

V$REPSENDER_SENT_LOG_COUNT_PARALLEL

This performance view displays the number of logs sent by the replication Sender, sorted by DML type. Whenever the number of replication logs specified for the REPLICATION_SENDER_LOG_COUNT_PERIOD property is sent, the data of this performance view is updated.

For parallel replication in EAGER mode, only the information about each Sender thread is displayed in this performance view; information about the Parent Sender is displayed in the V$REPSENDER_SENT_LOG_COUNT_PARALLEL performance view.

Column name	Type	Description
REP_NAME	VARCHAR(40)	This is the name of the replication object.
CURRENT_TYPE	VARCHAR(9)	This is the type of the replication Sender thread.
PARALLEL_ID	INTEGER	This is the identifier of one of several threads working in parallel for one Sender.
TABLE_OID	BIGINT	This is the table object identifier.
INSERT_LOG_COUNT	INTEGER	This is the number of INSERT logs sent.
DELETE_LOG_COUNT	INTEGER	This is the number of DELETE logs sent.
UPDATE_LOG_COUNT	INTEGER	This is the number of UPDATE logs sent.
LOB_LOG_COUNT	INTEGER	This is the number of LOB-related logs sent.

Column Information

REP_NAME

This is the name of the replication object created on the local server.

CURRENT_TYPE

Please refer to the description of the CURRENT_TYPE column in the V$REPGAP_PARALLEL performance view.

PARALLEL_ID

This is the identifier of one of several threads working in parallel for one Sender.

V$REPSENDER_STATISTICS

This view shows statistical information about the time that it takes for replication Senders to perform various tasks. When the TIMED_STATISTICS property is set to 1, cumulative statistics are maintained in this view. The interval at which this statistical information is updated is determined by the TIMER_THREAD_RESOLUTION and TIMER_RUNNING_LEVEL properties.

Column name	Type	Description
REP_NAME	VARCHAR(40)	This is the name of the replication object.
PARALLEL_ID	INTEGER	This is the identifier of one of several replication Sender threads working in parallel.
WAIT_NEW_LOG	BIGINT	This is the cumulative amount of time spent waiting for new logs to be written to the log buffer or log files.
READ_LOG_FROM_REPLBUFFER	BIGINT	This is the cumulative amount of time taken to read logs from the replication log buffer.
READ_LOG_FROM_FILE	BIGINT	This is the cumulative amount of time taken to read logs from log files.
CHECK_USEFUL_LOG	BIGINT	This is the cumulative amount of time taken to determine whether logs must be sent for replication.
ANALYZE_LOG	BIGINT	This is the cumulative amount of time taken to analyze logs and convert them into XLogs.
SEND_XLOG	BIGINT	This is the cumulative amount of time taken to send XLogs to Receiver Thread(s).

Column name	Type	Description
RECV_ACK	BIGINT	This is the cumulative amount of time spent waiting for and receiving ACK from Receiver Thread(s).
SET_ACKEDVALUE	BIGINT	This is the cumulative amount of time spent analyzing ACK values received from Receiver Thread(s).

Column Information

REP_NAME

This is the name of the replication object on the local server.

PARALLEL_ID

This is the identifier of one of several replicationSender threads having the same replication name. When parallel Sender threads are working in eager mode, a unique ID is given to each thread.

WAIT_NEW_LOG

This is the cumulative amount of time spent waiting for new logs to be written to the log buffer or log files. The Sender thread(s) reads these logs in order to send them to the Receiver thread(s).

READ_LOG_FROM_REPLBUFFER

This is the cumulative amount of time taken to read logs from the replication log buffer. This value is meaningful only when the REPLICATION_LOG_BUFFER_SIZE property is set to a value greater than 0.

READ_LOG_FROM_FILE

This is the cumulative amount of time taken to read logs from log files.

CHECK_USEFUL_LOG

This is the cumulative amount of time taken to determine whether logs must be sent for replication.

ANALYZE_LOG

This is the cumulative amount of time taken to analyze logs and convert them into XLogs.

SEND_XLOG

This is the cumulative amount of time taken to send XLogs to Receiver Thread(s).

RECV_ACK

This is the cumulative amount of time spent waiting for ACK and receiving ACK from Receiver Thread(s).

SET_ACKEDVALUE

This is the cumulative amount of time spent analyzing ACK values received from Receiver Thread(s).

V$REPSENDER_TRANSTBL

This view displays information about the replication Sender's transaction table.

Column name	Type	Description
REP_NAME	VARCHAR(40)	The name of the replication object
START_FLAG	BIGINT	A flag indicating startup options
LOCAL_TID	BIGINT	The local transaction identifier
REMOTE_TID	BIGINT	The remote transaction identifier
BEGIN_FLAG	INTEGER	Indicates whether 'BEGIN' acknowledgement has been sent
BEGIN_SN	BIGINT	The first log record sequence number of the transaction

Column Information

REP_NAME

This is the name of the replication object on the local server.

START_FLAG

Please refer to the description of the START_FLAG column in the V$REPSENDER performance view.

LOCAL_TID

This is the identifier of the transaction being executed on the local server.

REMOTE_TID

This is the identifier of the transaction being executed on the remote server.

V$REPSENDER_TRANSTBL_PARALLEL

This view displays information about transaction tables used by multiple replication Sender threads working in parallel.

Column name	Type	Description
REP_NAME	VARCHAR(40)	The name of the replication object
CURRENT_TYPE	VARCHAR(9)	The type of the replication Sender thread
LOCAL_TID	BIGINT	The local transaction identifier
REMOTE_TID	BIGINT	The remote transaction identifier

Column name	Type	Description
BEGIN_FLAG	INTEGER	Indicates whether 'BEGIN' acknowledgement has been sent
BEGIN_SN	BIGINT	The first log record sequence number of the transaction
PARALLEL_ID	INTEGER	The identifier of one of several replication Senders working in parallel

Column Information

REP_NAME

This is the name of the replication object.

CURRENT_TYPE

Please refer to the description of the CURRENT_TYPE column in the V$REPGAP_PARALLEL performance view.

LOCAL_TID

This is the identifier of the transaction being executed on the local server.

REMOTE_TID

This is the identifier of the transaction being executed on the remote server.

PARALLEL_ID

This is the identifier of one of several replication Sender threads working in parallel.

V$REPSYNC

This view displays information about tables that are synchronized using replication.

Column name	Type	Description
REP_NAME	VARCHAR(40)	The name of the replication object
SYNC_TABLE	VARCHAR(128)	The name of the table to be synchronized
SYNC_PARTITION	VARCHAR(128)	The name of the partition to be synchronized
SYNC_RECORD_COUNT	BIGINT	The number of records that have been synchronized on the remote server
SYNC_SN	BIGINT	Not currently used

Column Information

REP_NAME

This is the name of the replication object on the local server.

SYNC_TABLE

This is the name of the table that is the target for synchronization.

SYNC_PARTITION

This is the name of the partition that is the target for synchronization.

SYNC_RECORD_COUNT

When data in replication target tables on the local server are synchronized with those on the remote server, the data are synchronized in batches of records, the size of which is specified in the REPLICATION_SYNC_TUPLE_COUNT property of Altibase.

While synchronization is underway, this is the number of records that have been synchronized. A value of -1 indicates that synchronization is complete.

V$RESERVED_WORDS

This view displays all the keywords supported by SQL statement.

Column name	Type	Description
KEYWORD	VARCHAR(40)	The name of the keyword.
LENGTH	INTEGER	The length of the keyword
RESERVED_TYPE	INTEGER	The type of the keyword.

Column Information

KEYWORD

This is the name of the keyword.

LENGTH

This is the length of the keyword.

RESERVED_TYPE

This is the type of the keyword.

- 0: This keyword cannot be used as column name.
- 1: This keyword can be used as column name.

V$SBUFFER_STAT

This view displays statistical information about secondary buffers.

Column name	Type	Description
PAGE_COUNT	INTEGER	The secondary buffer size (the number of pages)
HASH_BUCKET_COUNT	INTEGER	The number of hash table buckets
HASH_CHAIN_LATCH_COUNT	INTEGER	The number of latches used in the hash table
CHECKPOINT_LIST_COUNT	INTEGER	The number of pages in checkpoint lists
HASH_PAGES	INTEGER	The number of pages inserted into the hash table
FLUSH_PAGES	INTEGER	The number of flushed pages
CHECKPOINT_LIST_PAGES	INTEGER	The number of pages in checkpoint lists
GET_PAGES	BIGINT	The number of page requests
READ_PAGES	BIGINT	The number of page reads
WRITE_PAGES	BIGINT	The number of page writes
HIT_RATIO	DOUBLE	The secondary buffer hit ratio
SINGLE_PAGE_READ_USEC	BIGINT	The single page read time
SINGLE_PAGE_WRITE_USEC	BIGINT	The single page write time
MPR_READ_USEC	BIGINT	The time spent on reading for full scan read
MPR_READ_PAGE_COUNT	BIGINT	The number of pages read for full scan read
SINGLE_READ_PERF	DOUBLE	The amount of single pages read(KB)/sec
MULTI_READ_PERF	DOUBLE	The amount of pages read by mpr(KB)/sec

Column Information

PAGE_COUNT

This displays the size of secondary buffers in the number of pages.

HASH_BUCKET_COUNT

This is the number of hash table buckets.

HASH_CHAIN_LATCH_COUNT

This is the number of chain latches used in the hash table.

CHECKPOINT_LIST_COUNT

This is the number of checkpoint lists.

HASH_PAGES

This is the number of pages inserted into the hash table. This value denotes the number of pages currently in use.

FLUSH_PAGES

This is the total number of pages flushed from secondary buffers, from server startup to the present.

CHECKPOINT_LIST_PAGES

This is the number of pages existing in checkpoint lists.

GET_PAGES

This is the cumulative number of times the buffer manager has requested for secondary buffer pages for the purpose of reading data, from server startup to the present.

READ_PAGES

This is the cumulative number of times the buffer manager has requested for secondary buffer pages for the purpose of reading data, from server startup to the present.

WRITE_PAGES

This is the cumulative number of times pages were written to secondary buffers.

HIT_RATIO

This is the cumulative hit ratio for secondary buffers, from server startup to the present.

SINGLE_PAGE_READ_USEC

This is the cumulative amount of time spent on reading one page from secondary buffers (unit: micro-seconds).

SINGLE_PAGE_WRITE_USEC

This is the cumulative time spent writing one page to the auxiliary buffer. (Unit: micro-seconds)

MPR_READ_USEC

This is the cumulative amount of time spent on writing one page to secondary buffers (unit: micro-seconds).

MPR_READ_PAGE_COUNT

This is the cumulative number of data pages read simultaneously from secondary buffers for the execution of a "full scan".

SINGLE_READ_PERF

This is the average number of bytes read per second when one data page is read from secondary buffers (unit: kB/sec).

MULTI_READ_PERF

This is the average number of bytes read per second when multiple data pages are read from secondary buffers for the execution of a "full scan" (unit: kB/sec).

V$SEGMENT

This view shows information about segments that make up disk tables and disk indexes, including their status, kind, and the index to which they are allocated.

Column name	Type	Description
SPACE_ID	INTEGER	The tablespace identifier
TABLE_OID	BIGINT	The object identifier of the table header
SEGMENT_PID	INTEGER	The identifier of the page in which the segment is stored
SEGMENT_TYPE	VARCHAR(7)	The type of segment
SEGMENT_STATE	VARCHAR(7)	The status of the segment
EXTENT_TOTAL_COUNT	BIGINT	The total number of extents assigned to the segment

Column Information

SEGMENT_PID

This is the identifier of the page in which the segment header is stored.

SEGMENT_TYPE

- INDEX: This indicates that the segment is an index segment.
- LOB: This indicates that the segment is an LOB segment.
- TABLE: This indicates that the segment is an table segment.
- TSSEG: This indicates that the segment is a TSS segment.
- UDSEG: This indicates that the segment is an undo segment.

SEGMENT_STATE

- USED: This indicates that the segment is being used.

- FREE: This indicates that the segment is empty.

EXTENT_TOTAL_COUNT

This is the total number of extents allocated to the segment.

V$SEQ

This view displays sequence-related information.

Column name	Type	Description
SEQ_OID	BIGINT	The object identifier of the sequence
CURRENT_SEQ	BIGINT	The current value of the sequence
START_SEQ	BIGINT	The starting value of the sequence
INCREMENT_SEQ	BIGINT	The increment value of the sequence
CACHE_SIZE	BIGINT	The size of the cache
MAX_SEQ	BIGINT	The maximum sequence value
MIN_SEQ	BIGINT	The minimum sequence value
IS_CYCLE	VARCHAR(7)	Indicates whether the sequence is cyclical

Column Information

SEQ_OID

This is a unique sequence identifier, which is assigned internally by the system when the sequence is created. It has the same value as a TABLE_OID in the SYS_TABLES_ meta table, for which the value in the TABLE_TYPE column will be 'S' (Sequence).

CURRENT_SEQ

This is the current sequence value.

START_SEQ

This is the sequence value that was specified when the sequence was first created.

INCREMENT_SEQ

This is value by which the sequence is incremented.

MAX_SEQ

This is the maximum value that can be generated using the sequence.

MIN_SEQ

This is the minimum value that can be generated using the sequence.

IS_CYCLE

When the sequence reaches its maximum possible value, this indicates whether the sequence will cycle and generate values starting from the minimum value again.

- The sequence cycles
- NO: The sequence does not cycle. If the sequence has reached the maximum possible value and an attempt is made to generate another sequence value, an error occurs.

V$SERVICE_THREAD

This view displays information about service threads.

Column name	Type	Description
ID	INTEGER	The service thread identifier
TYPE	VARCHAR(20)	The service thread access method
STATE	VARCHAR(10)	The current status of the service thread
RUN_MODE	VARCHAR(9)	The mode of execution of the service thread
SESSION_ID	BIGINT	The identifier of the session in which the service thread is executed
STATEMENT_ID	INTEGER	The identifier of the statement being executed by the service thread
START_TIME	INTEGER	The time at which the service thread was created
EXECUTE_TIME	BIGINT	The time taken for the service thread to execute a query
TASK_COUNT	INTEGER	The number of sessions being handled by the service thread
READY_TASK_COUNT	INTEGER	The number of sessions waiting for service threads to process their requests
THREAD_ID	BIGINT	The thread ID of the service thread

A thread in server process that receives and fulfills request(queries) from clients is called a service thread. Altibase provides the following two modes for the creation of service threads:

- Dedicated Thread Mode:
  When a multiple number of clients connect to the server and execute queries, one service thread is created for each client session in order to execute queries.

- Multiplexing Thread Mode:
  The Altibase server creates only the number of service threads optimized for the server and client sessions share these.

Altibase is designed to always maintain the optimal number of service threads by dynamically adding or deleting service threads as needed; however, the minimum number of service threads specified in the DEDICATED_THREAD_INIT_COUNT or MULTIPLEXING_THREAD_COUNT property is maintained.

Column Information

ID

This is the identifier of the service thread. This is an identifier that is managed within Altibase, rather than a system thread identifier (that is, a Light Weight Process ID).

TYPE

This shows the service thread connection method. It can have the following values:

- SOCKET (MULTIPLEXING): Connection via TCP or Unix Domain (UDP)

- SOCKET (DEDICATED): Connection via TCP or Unix Domain Socket

- IPC: Connection via IPC

- IPCDA: Connection via IPCDA

STATE

This indicates the current status of the service thread. It can have the following values:

- NONE: The service thread is being initialized.

- POLL: The service thread is waiting for an event.

- QUEUE-WAIT: The service thread is waiting in the queue.

- EXECUTE: The service thread is executing a statement.

- UNKNOWN: The status of the service thread cannot be determined.

RUN_MODE

This shows the mode of execution of the service thread. It can be either SHARED or DEDICATED.

- SHARED: The service thread run in multiplexing mode, and one service thread serves multiple client connections.

- DEDICATED: One client connection is allocated to one service thread, and uses the thread exclusively.

Switching the operating mode of a service thread is currently possible only for queue-related tasks. The mode can only be switched from SHARED mode to DEDICATED mode.

STATEMENT_ID

This is the identifier of the SQL statement that is currently being executed by the service thread.

START_TIME

This is the time point at which the service thread was created. It is based on system time. (Unit: seconds)

EXECUTE_TIME

This is the amount of time that the service thread has taken to execute the current query. (Unit: microseconds)

TASK_COUNT

This is the total number of sessions that are assigned to the service thread.

READY_TASK_COUNT

This is the number of sessions that are waiting for their requests to be processed by service threads

V$SERVICE_THREAD_MGR

This view displays the number of added and removed service threads.

Column name	Type	Description
ADD_THR_COUNT	INTEGER	The accumulated number of times that service threads has been added
REMOVE_THR_COUNT	INTEGER	The accumulated number of times that service threads has been removed

Altibase always maintains the optimal number of service threads by dynamically adding or deleting service threads as needed; this performance view displays the accumulated number of times service threads have been added and removed.

Column Information

ADD_THR_COUNT

This is the accumulated number of times that service threads have been added.

REMOVE_THR_COUNT

This is the accumulated number of times that service threads have been removed.

V$SESSION

The V$SESSION represents information on a client session created in Altibase.

Column name	Type	Description
ID	BIGINT	The session identifier

Column name	Type	Description
TRANS_ID	BIGINT	The identifier of the transaction currently being executed in the session
TASK_STATE	VARCHAR(11)	The task status
COMM_NAME	VARCHAR(64)	Connection information
XA_SESSION_FLAG	INTEGER	The XA session flag
XA_ASSOCIATE_FLAG	INTEGER	The XA associate flag
QUERY_TIME_LIMIT	BIGINT	See below
DDL_TIME_LIMIT	BIGINT	See below
FETCH_TIME_LIMIT	BIGINT	See below
UTRANS_TIME_LIMIT	BIGINT	See below
IDLE_TIME_LIMIT	BIGINT	See below
IDLE_START_TIME	INTEGER	See below
ACTIVE_FLAG	INTEGER	The active transaction flag
OPENED_STMT_COUNT	INTEGER	The number of opened statements
CLIENT_PACKAGE_VERSION	VARCHAR(40)	The client package version
CLIENT_PROTOCOL_VERSION	VARCHAR(40)	The client communication protocol version
CLIENT_PID	BIGINT	The client process ID
CLIENT_TYPE	VARCHAR(40)	The type of the client
CLIENT_APP_INFO	VARCHAR(128)	The type of the client application
CLIENT_NLS	VARCHAR(40)	The client character set
DB_USERNAME	VARCHAR(128)	The database user name
DB_USERID	INTEGER	The database user identifier
DEFAULT_TBSID	BIGINT	The user's default tablespace identifier
DEFAULT_TEMP_TBSID	BIGINT	The user's default temporary tablespace identifier
SYSDBA_FLAG	INTEGER	Indicates whether the connection was made as sysdba
AUTOCOMMIT_FLAG	INTEGER	The autocommit flag
SESSION_STATE	VARCHAR(13)	The status of the session
ISOLATION_LEVEL	INTEGER	The isolation level
REPLICATION_MODE	INTEGER	The replication mode
TRANSACTION_MODE	INTEGER	The transaction mode
COMMIT_WRITE_WAIT_MODE	INTEGER	See below
OPTIMIZER_MODE	INTEGER	The optimization mode
HEADER_DISPLAY_MODE	INTEGER	Indicates whether only the column names are output, or whether the table names are output along with the column names when the results of a SELECT query are output. 0: The table names are displayed along with the column names. 1: Only the column names are output.

Column name	Type	Description
CURRENT_STMT_ID	INTEGER	The identifier of the current statement
STACK_SIZE	INTEGER	The size of the stack (Unit: bytes)
DEFAULT_DATE_FORMAT	VARCHAR(64)	The default date format e.g.) DD-MON-RRRR
TRX_UPDATE_MAX_LOGSIZE	BIGINT	The maximum size of the DML Log (Unit: bytes)
PARALLE_DML_MODE	INTEGER	Deprecated
LOGIN_TIME	INTEGER	The amount of time the client has been logged in
FAILOVER_SOURCE	VARCHAR(256)	The kind of Fail-Over and information about the connection for which Fail-Over was conducted
NLS_TERRITORY	VARCHAR(40)	The territory name of the session
NLS_ISO_CURRENCY	VARCHAR(40)	The ISO currency symbol of the session
NLS_CURRENCY	VARCHAR(10)	The local currency symbol of the session
NLS_NUMERIC_CHARACTERS	VARCHAR(2)	The decimal character and group separator of the session
TIME_ZONE	VARCHAR(40)	The region name, abbreviation or UTC offset value of the time zone set for the session.
LOB_CACHE_THRESHOLD	INTEGER	The value set for the LOB_CACHE_THRESHOLD property
QUERY_REWRITE_ENABLE	VARCHAR(7)	The value set for the QUERY_REWRITE_ENABLE property
DBLINK_GLOBAL_TRANSACTION_LEVEL	INTEGER	The value set for the DBLINK_GLOBAL_TRANSACTION_LEV EL property
DBLINK_REMOTE_STATEMENT_AUTOCOMMIT	INTEGER	The value set for the DBLINK_REMOTE_STATEMENT_AUTO COMMIT property
MAX_STATEMENTS_PER_SESSION	INTEGER	The maximum number of statements that are allowed in session.
SSL_CIPHER	VARCHAR(256)	The cipher algorithm currently being used
SSL_CERTIFICATE_SUBJECT	VARCHAR(256)	Client certificate information
SSL_CERTIFICATE_ISSUER	VARCHAR(256)	The certificate authority that signed the client certificate
CLIENT_INFO	VARCHAR(128)	The information of accessed client application
MODULE	VARCHAR(128)	The name of a module in a procedure which is currently being executed.
ACTION	VARCHAR(128)	The action status of a procedure currently being executed.
REPLICATION_DDL_SYNC	INTEGER	Wheter to replicate DDL during replication
REPLICATION_DDL_TIMELIMIT	BIGINT	See below
MESSAGE_CALLBACK	VARCHAR(7)	The client message callback registration status

Column Information

ID

This is the unique identifier of a currently connected session.

TRANS_ID

This is the identifier of the transaction currently being executed in the session. If no transaction is currently underway, the value of -1 will be shown.

TASK_STATE

This indicates the status of the current task. It can have the following values:

STATE	Description
WAITING	The state in which the task is waiting for a request from a client
READY	The state in which the task has been received from a client and is waiting for a thread to be assigned to it
EXECUTING	The state in which a thread has been assigned to the task and is processing it
QUEUE WAIT	The state in which the task is waiting to be queued. After the task is queued, it is eventually dequeued.
QUEUE READY	The state in which the task has been queued. It will be dequeued once a thread has been assigned to it.
UNKNOWN	The state of the task cannot be determined.

COMM_NAME

This is the client connection information, the format of which varies depending on which communication protocol (TCP/IP, UNIX domain sockets, IPC, IPCDA or SSL) is used. In the case of TCP/IP and SSL, this information also includes the client IP address and port number.

XA_SESSION_FLAG

Indicates whether the current session is an XA session.

- 0: NON-XA (not an XA session)

XA_ASSOCIATE_FLAG

This shows the state of association between the session and the global transaction.

QUERY_TIME_LIMIT

This is the query timeout value for the current session.

DDL_TIME_LIMIT

This is the timeout value for DDL statements for the current session.

FETCH_TIME_LIMIT

This is the fetch timeout value for the current session.

UTRANS_TIME_LIMIT

This is the update transaction timeout value for the current session.

IDLE_TIME_LIMIT

This is the idle timeout value for the current session.

IDLE_START_TIME

This shows the time at which the session entered an Idle state.

ACTIVE_FLAG

If the session is executing a statement, the value of 1 is shown. However, if the session has merely connected, or has finished committing or rolling back a transaction, a value of 0 will be shown.

OPENED_STMT_COUNT

This shows the number of statements that are currently being executed by the session.

CLIENT_PACKAGE_VERSION

This is the package version of the connected client.

CLIENT_PROTOCOL _VERSION

This is the communication protocol version being used by the connected client.

CLIENT_PID

This is the process ID of the connected client. This value is not available for Java applications.

CLIENT_TYPE

This is a string that indicates the type of the connected client.

It consists of the following:

```
CLIENT_TYPE ::= app-type hypen word-size endian
    app-type  ::= CLI | WIN_ODBC | UNIX_ODBC
    hypen     ::= -
    word-size ::= 32 | 64
    endian    ::= BE | LE
BE : Big Endian, LE : Little Endian
```

Example)

```
CLI-32LE
UNIX_ODBC-32BE
```

CLIENT_APP_INFO

This is information about the connected client application. This information is set by the client application.

CLIENT_NLS

This indicates the character set in use on the connected client.

DB_USERNAME

This is the name of the database user being used on the connected client.

DB_USERID

This is a numerically expressed user identifier, used by Altibase to distinguish between users.

DEFAULT_TBSID

This is the identifier of the default tablespace for the user.

DEFAULT_TEMP_TBSID

This is the identifier of the default temporary tablespace for the user.

SYSDBA_FLAG

This indicates whether or not the session is connected in sysdba mode.

- 1: sysdba mode

AUTOCOMMIT_FLAG

This indicates whether AUTOCOMMIT is active for the connected session.

- 0: non-autocommit
- 1: autocommit

SESSION_STATE

STATE	Description
INIT	The state in which the session is waiting for a request from a client
AUTH	The state in which user authorization is complete
SERVICE READY	The state in which service is ready (The state "A transaction cannot be created" is returned only for XA sessions.)
SERVICE	The service state
END	The state of normal completion (COMMIT in the case where there is a transaction)

STATE	Description
ROLLBACK	The state of abnormal termination (ROLLBACK in the case where there is a transaction) This occurs when a client is disconnected or a server forcibly disconnects a session.
UNKNOWN	The state cannot be determined

ISOLATION_LEVEL

This indicates the isolation level for the session.

REPLICATION_MODE

This indicates the replication mode for the session.

- 0: DEFAULT(Réplication)
- 16: NONE

TRANSACTION_MODE

This indicates the transaction mode for the session.

- 0: READ/WRITE
- 4: READ ONLY

COMMIT_WRITE_WAIT_MODE

- 0: When a transaction is committed, do not wait until the logs are written to disk.
- 1: When a transaction is committed, wait until the logs are written to disk.

OPTIMIZER_MODE

This indicates the optimization mode that has been set for the session.

- 1: the optimization mode is rule-based
- 0: the optimization mode is cost-baseCURRENT_STMT_ID

This indicates the identifier of the statement currently being executed.

STACK_SIZE

This is the size of the stack for the query processor that has been set for the current session.

DEFAULT_DATE_FORMAT

This is the default date format that has been set for the session. (Please refer to the description of the Datetime data type in Chapter1: Data Types.)

Example)

```
DD-MON-RRRR
```

TRX_UPDATE_MAX_LOGSIZE

This is the maximum size of logs that can be generated by a single DML statement.

LOGIN_TIME

This indicates the amount of time that the client has been logged in.

FAILOVER_SOURCE

This value indicates the kind of Fail-Over that occurred, as well as the connection properties for the server related to which Fail-Over was performed. "Connection properties" means, in the case of CTF (Connection Time Fail-Over), the IP address and port number of the database server to which a connection attempt was first made, and, in the case of STF (Service Time Failover), the IP address and port number of the database server with which a connection was interrupted.

Example) When the connection properties of the Active (primary) Server are 127.0.0.1:10000 and the connection properties of the Standby (secondary) Server are 127.0.0.2:20000:

- If a CTF occurs after failure to connect to 127.0.0.1 and connects to 127.0.0.2, the value of FAILOVER_SOURCE is as follows: CTF 127.0.0.1:10000

- If it is connected to 127.0.0.2 but got an error and got STF with 127.0.0.1, the value of FAILOVER_SOURCE is as follows: STF 127.0.0.2:20000

NLS_TERRITORY

This displays the territory name of the currently connected session.

NLS_ISO_CURRENCY

This displays the ISO currency symbol of the currently connected session.

NLS_CURRENCY

This displays the local currency symbol of the currently connected session.

NLS_NUMERIC_CHARACTERS

This displays the decimal character and group separator of the currently connected session.

TIME_ZONE

The region name, abbreviation or UTC offset value of the time zone set for the session.

LOB_CACHE_THRESHOLD

This indicates the value set for the LOB_CACHE_THRESHOLD property in the session. For further information about the LOB_CACHE_THRESHOLD property, please refer to Chapter 2.

QUERY_REWRITE_ENABLE

This indicates the value set for the QUERY_REWRITE_ENABLE property in the session. For more detailed information about the QUERY_REWRITE_ENABLE property, please refer to Chapter 2.

- FALSE: The function-based indexes are not applied when rewriting a query on the Altibase server (disable).

- TRUE: The function-based indexes are applied when rewriting a query on the Altibase server (enable).

DBLINK_GLOBAL_TRANSACTION_LEVEL

This indicates the execution level of global transactions set for the DBLINK_GLOBAL_TRANSACTION_LEVEL property in the session. For further information about the DBLINK_GLOBAL_TRANSACTION_LEVEL property, please refer to Chapter 2.

- 0: Remote statement execution level
- 1: Simple transaction commit level
- 2: Two-Phase Commit

DBLINK_REMOTE_STATEMENT_AUTOCOMMIT

This indicates the AUTOCOMMIT mode of the remote database set for the DBLINK_REMOTE_STATEMENT_AUTOCOMMIT property in the session. For further information about the DBLINK_REMOTE_STATEMENT_AUTOCOMMIT property, please refer to Chapter 2.

- 0: autocommit-off
- 1: autocommit-on

MAX_STATEMENTS_PER_SESSION

The value of the MAX_STATEMENTS_PER_SESSION is the default value.

SSL_CERTIFICATE_SUBJECT

This is not displayed if client authentication has been ommitted (i.e. if the value of the SSL_CLIENT_AUTHENTICATION property is 0).

SSL_CERTIFICATE_ISSUER

This is not displayed if client authentication has been ommitted (i.e. if the value of the SSL_CLIENT_AUTHENTICATION property is 0).

CLIENT_INFO

The CLIENT_INFO is the information of accessed client application, and the value is configured by the client application program.

MODULE

This is information on the name of a module in a procedure which is currently being executed. Configuration can be arranged through the built-in procedure SET_MODULE( ).

ACTION

This is information on the action status of a procedure currently being executed.Configuration can be arranged through the built-in procedure SET_MODULE( ).

REPLICATION_DDL_SYNC

This indicates whether to allow DDL replication.

- 0: DDL replication is not supported during replicaiton
- 1: DDL replicaiton is supported during replication.

REPLICATION_DDL_TIMEOUT

This indicates DDL replication execution timeout value through replication of the current session.

The excess value is measured based on the local server performing DDL replicaiton.

MESSAGE_CALLBACK

This indicates the message callback registration status of the conncted client.

Depending on the message callback registration status, the server decides whether to send the message.

- REG
  The client registers a message callback, and the server sends a message to the client.
- UNREG
  The client has not registered a message callback, and the server does not send a message to the client.
- UNKNOWN
  If it is unknown whether the client has client registered a message callback, and the server sends the message to the client. If an older client without this feature connects, it will have in the UNKNOWN state.

V$SESSION_EVENT

This view shows cumulative statistical wait information about all wait events for each session that is currently connected to an Altibase.

Column name	Type	Description
SID	INTEGER	The identifier of the session
EVENT	VARCHAR(128)	The name of the wait event
TOTAL_WAITS	BIGINT	The total number of waits related to the wait event
TOTAL_TIMEOUTS	BIGINT	The total number of times that a resource could not be accessed after the specified time
TIME_WAITED	BIGINT	The total amount of time spent waiting for the wait event (in milliseconds)
AVERAGE_WAIT	BIGINT	The average amount of time spent waiting for the wait event (in milliseconds)

Column name	Type	Description
MAX_WAIT	BIGINT	The maximum time spent waiting for the wait event (in milliseconds)
TIME_WAITED_MICRO	BIGINT	The total amount of time spent waiting for the wait event (in microseconds)
EVENT_ID	INTEGER	The identifier of the wait event
WAIT_CLASS_ID	INTEGER	The identifier of the class of the wait event
WAIT_CLASS	VARCHAR(128)	The name of the class of the wait event

Column Information

SID

This is the identifier of a waiting session.

EVENT

This is the name of the wait event.

TOTAL_WAITS

This is the total number of waits related to the wait event.

TOTAL_TIMEOUTS

This is the number of failures to gain access to the requested resource even after the specified time has elapsed.

TIME_WAITED

This is the total time spent waiting for this wait event (in milliseconds).

AVERAGE_WAIT

This is the average amount of time spent waiting for the wait event (in milliseconds).

MAX_WAIT

This is the maximum time spent waiting for the wait event (in milliseconds).

TIME_WAITED_MICRO

This is the total amount of time spent waiting for this wait event (in microseconds).

EVENT_ID

This is the identifier of the wait event.

WAIT_CLASS_ID

This is the identifier of the wait class in which the wait event is classified.

WAIT_CLASS

This is the name of the class in which the wait event is classified.

V$SESSION_WAIT

This view displays information about wait events for all currently connected sessions. This view does not provide Information about wait events related to sessions that are no longer connected.

Column name	Type	Description
SID	BIGINT	The identifier of the session
SEQNUM	INTEGER	The identifier of the wait event
EVENT	VARCHAR(128)	The name of the wait event
P1	BIGINT	Parameter 1 of the wait event
P2	BIGINT	Parameter 2 of the wait event
P3	BIGINT	Parameter 3 of the wait event
WAIT_CLASS_ID	INTEGER	The identifier of the wait class
WAIT_CLASS	VARCHAR(128)	The name of the wait class
WAIT_TIME	BIGINT	The time spent waiting (in milliseconds)
SECOND_IN_WAIT	BIGINT	The time spent waiting (in seconds)

Column Information

SID

This is the identifier of a currently connected session.

SEQNUM

This is the identifier of the wait event associated with the session.

EVENT

This is the name of the wait event.

WAIT_CLASS_ID

This is the identifier of the class of the wait event.

WAIT_CLASS

This is the name of the wait class.

WAIT_TIME

This is the amount of time spent waiting for the wait event (in milliseconds).

SECOND_IN_WAIT

This is the amount of time spent waiting for the wait event (in seconds).

V$SESSION_WAIT_CLASS

This view shows cumulative statistical information about waits, classified according to session and wait event, for all currently connected sessions. This view does not provide information about wait events related to sessions that are no longer connected.

Column name	Type	Description
SID	INTEGER	The session identifier
SERIAL	INTEGER	The identifier of the wait event
WAIT_CLASS_ID	INTEGER	The identifier of the wait class
WAIT_CLASS	VARCHAR(128)	The name of the wait class
TOTAL_WAITS	BIGINT	The total number of waits for this wait event in this session
TIME_WAITED	DOUBLE	The total amount of time waited for this wait event in this session (in milliseconds)

Column Information

SID

This is the identifier of the session.

SERIAL

This is the identifier of the wait event.

WAIT_CLASS_ID

This is the identifier of the wait class.

WAIT_CLASS

This is the name of the wait class.

TOTAL_WAITS

This is the total number of waits for this wait event in this session.

TIME_WAITED

This is the total time (in milliseconds) spent waiting for this wait event in this session.

Example

<Example> The following SELECT query outputs the total number of waits and the total amount of time spent waiting for each wait event in each session, classified by session, wait event and wait class.

```
select sid, serial, wait_class_id, sum(total_waits), sum(time_waited)
from v$session_wait_class
group by sid, serial, wait_class_id
order by total_waits desc;
```

V$SESSIONMGR

This view displays statistical information about sessions.

Column name	Type	Description
TASK_COUNT	INTEGER	The number of connected sessions
BASE_TIME	INTEGER	The current time
LOGIN_TIMEOUT_COUNT	INTEGER	See below
IDLE_TIMEOUT_COUNT	INTEGER	See below
QUERY_TIMEOUT_COUNT	INTEGER	See below
DDL_TIMEOUT_COUNT	INTEGER	See below
FETCH_TIMEOUT_COUNT	INTEGER	See below
UTRANS_TIMEOUT_COUNT	INTEGER	See below
SESSION_TERMINATE_COUNT	INTEGER	See below

Column Information

TASK_COUNT

This is the total number of currently connected sessions.

BASE_TIME

This is the current time, expressed in seconds.

LOGIN_TIMEOUT_COUNT

This is the number of login timeouts that have occurred since Altibase was started.

IDLE_TIMEOUT_COUNT

This is the number of idle timeouts that have occurred since Altibase was started.

DDL_TIMEOUT_COUNT

This is the number of times that DDL statements have timed out since Altibase was started.

QUERY_TIMEOUT_COUNT

This is the number of query timeouts that have occurred since Altibase was started.

FETCH_TIMEOUT_COUNT

This is the number of fetch timeouts that have occurred since Altibase was started.

UTRANS_TIMEOUT_COUNT

This is the number of UPDATE transaction timeouts that have occurred since Altibase was started.

SESSION_TERMINATE_COUNT

This is the number of sessions that have been forcibly disconnected by the sysdba since Altibase was started.

V$SESSTAT

This view shows statistics for all currently connected sessions.

Column name	Type	Description
SID	INTEGER	The identifier of the session.
SEQNUM	INTEGER	The serial number of each statistic
NAME	VARCHAR(128)	The name of the statistic
VALUE	BIGINT	The value returned for the statistic

For information about each status, please refer to V$STATNAME.

Column Information

SID

This is the unique identifier for the session.

SEQNUM

This is a serial number for the statistic.

NAME

This is the name of the statistic.

VALUE

This is the value returned for the statistic, expressed as a 64-bit integer.

V$SFLUSHER

This view displays information about tasks flushing secondary buffer pages to disk.

Column name	Type	Description
ID	INTEGER	The flusher identifier
ALIVE	INTEGER	Whether or not the flusher is currently active
CURRENT_JOB	INTEGER	The current job being performed: 1: Replacement flushing 2: Checkpoint flushing 3: Object flushing
DOING_IO	INTEGER	Whether or not the flusher is performing disk I/O
INIOB_COUNT	INTEGER	The number of times an internal buffer has been directly accessed in order to save contents to be flushed therein
REPLACE_FLUSH_JOBS	BIGINT	The cumulative number of completed replacement flushing tasks
REPLACE_FLUSH_PAGES	BIGINT	The cumulative number of pages written to disk by replacement flushing
REPLACE_SKIP_PAGES	BIGINT	The cumulative number of pages for which flushing was canceled during replacement flushing
CHECKPOINT_FLUSH_JOBS	BIGINT	The cumulative number of completed checkpoint flushing tasks
CHECKPOINT_FLUSH_PAGES	BIGINT	The cumulative number of pages written to disk by checkpoint flushing
CHECKPOINT_SKIP_PAGES	BIGINT	The cumulative number of pages for which flushing was canceled during checkpoint flushing
OBJECT_FLUSH_JOBS	BIGINT	The cumulative number of times object flushing has been performed
OBJECT_FLUSH_PAGES	BIGINT	The cumulative number of pages written to disk by object flushing

Column name	Type	Description
OBJECT_SKIP_PAGES	BIGINT	The cumulative number of pages for which flushing was canceled during object flushing
LAST_SLEEP_SEC	INTEGER	This is the length of time the flusher has slept after having completed all of its tasks
TIMEOUT	BIGINT	The number of times the sleeping flusher has woken up to check whether or not it has any tasks
SIGNALED	BIGINT	The number of times the flusher has been woken up by a signal from the Altibase server
TOTAL_SLEEP_SEC	BIGINT	The total length of time the flusher has slept
TOTAL_FLUSH_PAGES	BIGINT	The cumulative number of flushed pages
TOTAL_DW_USEC	BIGINT	The cumulative amount of time spent on writing the contents of doublewrite buffers to disk
TOTAL_WRITE_USEC	BIGINT	The cumulative amount of time spent on writing temporary pages to temporary files
TOTAL_SYNC_USEC	BIGINT	The cumulative amount of time spent on forcefully flushing data pages to disk
TOTAL_FLUSH_TEMP_PAGES	BIGINT	The cumulative number of flushed temporary pages
TOTAL_TEMP_WRITE_USEC	BIGINT	The cumulative amount of time spent on writing temporary pages to temporary files
DB_WRITE_PERF	DOUBLE	The average number of bytes per second for writing data pages to data files
TEMP_WRITE_PERF	DOUBLE	The average number of bytes per second for writing temporary pages to temporary files

Column Information

ID

This is the flusher identifier; this value is not a duplicate.

ALIVE

This indicates whether or not the flusher is currently active. Each flusher can be started or stopped with DCL statements.

CURRENT_JOB

This indicates the type of job the flusher is currently performing.

- 1: The flusher is performing replacement flushing. The purpose of replacement flushing is to flush buffers that have not been accessed for a long time to enable their replacement.
- 2: The flusher is performing checkpoint flushing. The purpose of checkpoint flushing is to flush buffers that have not been flushed for the longest time to shorten checkpoint time.
- 3: The flusher is performing object flushing on a certain object, such as an index, a table, a segment, and etc

DOING_IO

This indicates whether or not the flusher is performing disk I/O for the execution of its current task.

INIOB_COUNT

To save pages to disk, the flusher stores the contents in an internal buffer (IOB). This value indicates the number of times the internal buffer has been directly accessed in order to save contents to be flushed therein.

REPLACE_FLUSH_JOBS

This is the cumulative number of times replacement flushing has been performed.

REPLACE_FLUSH_PAGES

This is the cumulative number of pages written to disk by replacement flushing.

REPLACE_SKIP_PAGES

This is the cumulative number of pages for which flushing was canceled during replacement flushing, due to either policy or efficiency issues.

CHECKPOINT_FLUSH_JOBS

This is the cumulative number of times checkpoint flushing has been performed.

CHECKPOINT_FLUSH_PAGES

This is the cumulative number of pages written to disk by checkpoint flushing.

CHECKPOINT_SKIP_PAGES

This is the cumulative number of pages for which flushing was canceled during checkpoint flushing, due to either policy or efficiency issues.

OBJECT_FLUSH_JOBS

This is the cumulative number of times object flushing has been performed.

OBJECT_FLUSH_PAGES

This is the cumulative number of pages written to disk by object flushing.

OBJECT_SKIP_PAGES

This is the cumulative number of pages for which flushing was canceled during object flushing, due to either policy or efficiency issues.

LAST_SLEEP_SEC

This is the length of time the flusher has most recently slept after having completed all of its tasks.

TIMEOUT

For flushers that go to sleep in the absence of tasks, it is necessary for them to wake up at regular intervals to check whether or not they have tasks. This value is the cumulative number of times the flusher has woken up.

SIGNALED

To enhance the performance of a certain task, Altibase can signal a sleeping flusher and wake it up. This value is the number of times that the flusher has been woken up by such a signal.

TOTAL_SLEEP_SEC

This is the total length of time that the flusher was asleep on standby in the absence of tasks.

TOTAL_FLUSH_PAGES

This is the cumulative number of pages that have been flushed in the course of checkpoint flushing, replacement flushing or object flushing.

TOTAL_DW_USEC

This is the cumulative amount of time taken to write the contents of doublewrite buffers to disk. To "doublewrite" is to write pages first to the DW file, called the doublewrite buffer, before writing them to data files. Once they have been written to the doublewrite buffer, they are then written to data files in the correct location. If the operating system crashes during the process of writing pages to data files, or the data files are corrupted, it is possible to perform data recovery using the uncorrupted pages in the doublewrite buffer.

TOTAL_WRITE_USEC

This is the cumulative amount of time spent on writing data pages to data files. This value does not include the amount of time spent flushing data to disk.

TOTAL_SYNC_USEC

This is the cumulative amount of time spent on forcefully flushing data to disk.

TOTAL_FLUSH_TEMP_PAGES

This is the cumulative number of flushed temporary pages (temporary pages are used for storing temporary tables, which are used for sort operations and hash joins).

TOTAL_TEMP_WRITE_USEC

This is the amount of time spent on writing temporary pages to temporary files.

DB_WRITE_PERF

This is the average number of bytes per second for writing data pages to data files (unit: kB/sec).

TEMP_WRITE_PERF

This is the average number of bytes per second for writing temporary pages to temporary files (unit: kB/sec).

V$SFLUSHINFO

This view displays flush information about secondary buffers.

Column name	Type	Description
FLUSHER_COUNT	INTEGER	The number of pages to be flushed
CHECKPOINT_LIST_COUNT	INTEGER	The number of checkpoint lists
REQ_JOB_COUNT	INTEGER	The number of tasks currently registered for the flush manager
REPLACE_PAGES	INTEGER	The number of pages to be flushed by replacement flushing
CHECKPOINT_PAGES	INTEGER	The number of pages to be flushed by checkpoint flushing
MIN_BCB_ID	INTEGER	The BCB identifier which corresponds to the checkpoint target page with the fastest recovery LSN
MIN_SPACEID	INTEGER	The tablespace ID of the checkpoint target page with the fastest recovery LSN
MIN_PAGEID	INTEGER	The page ID of the checkpoint target page with the fastest recovery LSN

Column Information

FLUSHER_COUNT

This is the number of pages to be flushed from the secondary buffer to disk.

CHECKPOINT_LIST_COUNT

This is the number of checkpoint lists.

REQ_JOB_COUNT

This is the number of tasks registered for the flush manager.

REPLACE_PAGES

This is the number of pages to be flushed by replacement flushing, from the secondary buffer to disk.

CHECKPOINT_PAGES

This is the number of pages to be flushed by checkpoint flushing, from the secondary buffer to disk.

MIN_BCB_ID

This is the BCB identifier which corresponds to the checkpoint target page with the fastest recover LSN.

MIN_SPACEID

This is the tablespace ID of the checkpoint target page with the fastest recovery LSN.

MIN_PAGEID

This is the page ID of the checkpoint target page with the fastest recovery LSN.

V$SNAPSHOT

The following table indicates the information of SNAPSHOT settings, usage information of memory and disk undo tablespace.

Column name	Type	Description
SCN	BIGINT	The SNAPSHOT SCN value specified in the snapshot
BEGIN_TIME	BIGINT	UNIX_TIME when specifying the snapshot settings.
BEGIN_MEM_USAGE	INTEGER	Memory usage ration when specifying the snapshot settings
BEGIN_DISK_UNDO_USAGE	INTEGER	The usage ratio of disk undo tablespace at BEGIN when specifying the snapshot settings
CURRENT_TIME	BIGINT	The current UNIX_TIME
CURRENT_MEM_USAGE	INTEGER	The current memory usage ratio

Column name	Type	Description
CURRENT_DISK_UNDO_USAGE	INTEGER	The usage ratio of current disk undo tablespace

Column Information

SCN

SCN indicates the SCN value specified when BEGIN SNAPSHOT settings were configured. iLoader executes EXPORT data based upon SCN.

BEGIN_TIME

The time when BEGING SNAPSHOT statement is executed is expressed with UNIX_TIME by BEGIN_TIME.

BEGIN_MEM_USAGE

The memory usage when BEGIN SNAPSHOT statement is implemented through the percentage.

BEGIN_DISK_UNDO_USAGE

The percentage of disk undo tablespace is displayed when the BEGING SNAPSHOT statement is executed.

CURRENT_TIME

The current time is expressed with UNIX_TIME.

CURRENT_MEM_USAGE

The current memory usage is displayed with the percentage.

CURRENT_DISK_UNDO_USAGE

The current usage of disk undo tablespace is displayed with the percentage.

V$SQLTEXT

This view displays information about SQL text that is currently being executed in the server.

Column name	Type	Description
SID	INTEGER	The identifier of the session
STMT_ID	INTEGER	The identifier of the statement
PIECE	INTEGER	The serial number of the text fragment
TEXT	VARCHAR(64)	A fragment of SQL text

Column Information

SID

This is a unique number identifying the session in which the SQL text is being executed.

STMT_ID

This is the serial number of the fragment of the SQL statement being executed in the session.

PIECE

The complete SQL statement that is being executed is divided into 64-byte fragments of text and saved. PIECE is a serial number that identifies each line of text, ascending from 0.

TEXT

This is the actual 64-byte fragment of text constituting part of the SQL statement.

V$SQL_PLAN_CACHE

This view shows the current status of the SQL Plan Cache along with some related statistical information.

Column name	Type	Description
MAX_CACHE_SIZE	BIGINT	The maximum size of the SQL Plan Cache (in bytes)
CURRENT_HOT_LRU_SIZE	BIGINT	The current size of the HOT area of an LRU list
CURRENT_COLD_LRU_SIZE	BIGINT	The current size of the COLD area of an LRU list
CURRENT_CACHE_SIZE	BIGINT	The current size of the SQL Plan Cache (in bytes)
CURRENT_CACHE_OBJ_COUNT	INTEGER	The number of PCO currently registered in the SQL Plan Cache
CACHE_HIT_COUNT	BIGINT	The usage count of PCO registered in SQL Plan Cache
CACHE_MISS_COUNT	BIGINT	The number of times PCO was not found while searching for plans in the SQL Plan Cache
CACHE_IN_FAIL_COUNT	BIGINT	The number of failures due to cache maxmium size constraint when inserting new PCO into SQL Plan Cache
CACHE_OUT_COUNT	BIGINT	The number of PCO removed from the SQL Plan Cache
CACHE_INSERTED_COUNT	BIGINT	The number of PCO added to the SQL Plan Cache

Column name	Type	Description
NONE_CACHE_SQL_TRY_COUNT	BIGINT	The number of attempts by non-cached statements such as DDL and DCL

Column Information

MAX_CACHE_SIZE

This is the maximum size of the SQL Plan Cache. To reduce or increase this maximum size, execute 'ALTER SYSTEM SET SQL_PLAN_CACHE_SIZE = '.

CURRENT_HOT_LRU_SIZE

PCOs on the SQL Plan Cache LRU list that are frequently referred to are managed in a HOT area, the size of which is expressed in bytes.

CURRENT_COLD_LRU_SIZE

PCOs on the SQL Plan Cache LRU list that are not frequently referred to are managed in a COLD area, the size of which is expressed in bytes.

CURRENT_CACHE_SIZE

This is the total size of PCOs that are currently in the SQL Plan Cache.

CURRENT_CACHE_OBJ_COUNT

This is the number of PCOs that are in the SQL Plan Cache.

CACHE_HIT_COUNT

This is the total number of times that PCOs in the SQL Plan Cache have been used.

CACHE_MISS_COUNT

This is the number of attempts to refer to PCOs that do not exist in the SQL Plan Cache.

CACHE_IN_FAIL_COUNT

This is the number of times that a PCO could not be inserted into the cache due to the maximum memory size restriction of the cache, although an attempt was made to delete or remove PCOs infrequently refererred from the cache.

CACHE_OUT_COUNT

This is the number of PCOs that were deleted from the SQL Plan Cache.

CACHE_INSERTED_COUNT

This is the number of PCOs that were added to the SQL Plan Cache.

NONE_CACHE_SQL_TRY_COUNT

This is the number of attempts to execute statements that do not affect the plan cache. These statements are usually DDL or DCL statements.

V$SQL_PLAN_CACHE_PCO

This view displays information about PCOs registered in the SQL Plan Cache.

PCO is an object that contains information about SQL statement, execution plan and plan environment. It inhances query efficiency by sharing the execution plan between the sessions when executing the statement. There are two types of PCO, which are Parent PCO and Child PCO.

Parent PCO

PCO that has information to compare two SQL statements and manage them. Each SQL statement has different Parent PCO.

Child PCO

PCO that manages plan environment, which affects the execution plan, to compare them. For the same SQL statement different execution plans can be generated due to different plan environment such as user, NLS(National Language Support), statistics. Child PCO stores information about plan environment, execution plan and size of execution plan when PCO was created. It requires Parent PCO and one Parent PCO can have multiple Child PCOs.

Column name	Type	Description
SQL_TEXT_ID	VARCHAR(64)	The identifier of Parent PCO
PCO_ID	INTEGER	The identifier of Child PCO
CREATE_REASON	VARCHAR(28)	The reason the PCO was created
HIT_COUNT	INTEGER	The number of times PCO has been referred to
REBUILD_COUNT	INTEGER	The number of times PCO has been rebuilt
PLAN_STATE	VARCHAR(17)	The plan state of PCO
LRU_REGION	VARCHAR(11)	The region of the plan in the LRU list, which can be HOT_REGION or COLD_REGION
PLAN_SIZE	INTEGER	The plan size of PCO
FIX_COUNT	INTEGER	The number of statements referencing the PCO
PLAN_CACHE_KEEP	VARCHAR(6)	The Keep state of plan cache object

Column Information

SQL_TEXT_ID

This is the identifier of the Parent PCO.

PCO_ID

This is the identifier of the Child PCO.

CREATE_REASON

This displays the reason why PCO was created and can have the following values:

- CREATE_BY_CACHE_MISS
  SQL Plan Cache was missing the required PCO.

- CREATE_BY_PLAN_INVALIATION
  PCO was found in the SQL Plan Cache during PREPARE stage, but was not valid.

- CREATE_BY_PLAN_TOO_OLD
  The change width of statistical information about objects to which the plan refers has exceeded the limit, or a DDL statement was executed

HIT_COUNT

This is the number of times PCO has been referred to.

REBUILD_COUNT

This is the number of times PCO has been recompiled.

PLAN_STATE

This is the plan state of the PCO and can have the following values:

- READY
  SQL statement, execution plan and plan environment have been assigned to PCO.

- OLD_PLAN
  Plan is not valid and will not be used in the future.

LRU_REGION

Hot-Cold LRU list is a data structure that manages PCO replacement policy. The size of SQL Plan Cache is fixed by SQL_PLAN_CACHE_SIZE, Altibase server's property, therefore only limited number of PCO can be registered. This column indicates which region PCO belongs to.

- HOT_REGION

  Frequently used PCO

- COLD_REGION

  Less frequently used PCO

PLAN_SIZE

This is the plan size of the PCO.

FIX_COUNT

This shows the number of statements referencing the PCO. When FIX_COUNT is 1 or more, no victim is selected.

PLAN_CACHE_KEEP

This indicates the keep status of PCO and can have following values:

- KEEP
  PLAN is kept and will not be selected for victims
- UNKEEP
  PLAN can be selected as victim with unkeep status

V$SQL_PLAN_CACHE_SQLTEXT

This view displays information about [Parent PCO](#).

Column name	Type	Description
SQL_TEXT_ID	VARCHAR(64)	The identifier of Parent PCO
SQL_TEXT	VARCHAR(16384)	The SQL statements
CHILD_PCO_COUNT	INTEGER	The number of Child PCOs Parent PCO currently has
CHILD_PCO_CREATE_COUNT	INTEGER	The number of Child PCOs that have been created until now in Parent PCO
PLAN_CACHE_KEEP	VARCHAR(6)	The keep status of the PCO corresponding to SQL_TEXT_ID

Column Information

SQL_TEXT_ID

This is the identifier of the Parent PCO. The first 4 digits indicate the number of the bucket in which the Parent PCO is stored. The following digits indicate the serial number of the SQL statement in the bucket.

SQL_TEXT

This is the SQL statement.

CHILD_PCO_COUNT

This is the number of Child PCOs that the Parent PCO currently has.

CHILD_PCO_CREATE_COUNT

This is the number of Child PCOs that have been created in the Parent PCO until now. New Child PCOs are created in the Parent PCO in the two following cases:

- SQL statement is identical with the existing PCO but the environment in which the plan was created has changed.

- Objects that existing PCO refers to have changed, or the change width of statistical information about objects to which the plan refers has exceeded the limit.

PLAN_CACHE_KEEP

This shows the keep status of plan cache object corresponding to SQL_TEXT_ID and can have the following values:

- KEEP
  PLAN is kept and is not selected as victim.

- UNKEEP
  PLAN can be selected as victim with unkeep status.

V$STABLE_MEM_DATAFILES

This view shows the complete file path of the data files in the database.

Column name	Type	Description
MEM_DATA_FILE	VARCHAR(4096)	The full path of the data file

Column Information

MEM_DATA_FILE

This is the full path of the data files in the database.

V$STATEMENT

This view shows information about the most recently executed query in each currently connected session.

Column name	Type	Description
ID	INTEGER	The identifier of the statement
PARENT_ID	INTEGER	The identifier of the parent statement
CURSOR_TYPE	INTEGER	The cursor type

Column name	Type	Description
SESSION_ID	INTEGER	The ID of the session to which the statement belongs
TX_ID	BIGINT	The identifier of the transaction
QUERY	VARCHAR(16384)	SQL string performed
LAST_QUERY_START_TIME	INTEGER	The start time of the most recent query
QUERY_START_TIME	INTEGER	The start time of the current query
FETCH_START_TIME	INTEGER	The start time of the current fetch
EXECUTE_STATE	VARCHAR(8)	The start time of the current fetch
FETCH_STATE	VARCHAR(12)	The fetch state of statement
ARRAY_FLAG	INTEGER	The array execution flag
ROW_NUMBER	INTEGER	The number of the current row
EXECUTE_FLAG	INTEGER	The execution flag
BEGIN_FLAG	INTEGER	A flag that shows whether the current statement is opened or not
TOTAL_TIME	BIGINT	The total elapsed time
PARSE_TIME	BIGINT	The time taken to parse the statement
VALIDATE_TIME	BIGINT	The time taken to validate the statement
OPTIMIZE_TIME	BIGINT	The time taken to optimize the statement
EXECUTE_TIME	BIGINT	The time taken to execute the statement
FETCH_TIME	BIGINT	The time taken to perform a fetch operation
SOFT_PREPARE_TIME	BIGINT	The time taken to search for a plan in the SQL Plan Cache during the Prepare process
SQL_CACHE_TEXT_ID	VARCHAR(64)	The identifier of Parent PCO or NO_SQL_CACHE_STMT
SQL_CACHE_PCO_ID	INTEGER	The identifier of Child PCO

Column name	Type	Description
OPTIMIZER	BIGINT	The optimization mode
COST	BIGINT	The optimization cost
USED_MEMORY	BIGINT	Reserved for future use
READ_PAGE	BIGINT	The optimization cost
WRITE_PAGE	BIGINT	The number of disk pages that have been written to
GET_PAGE	BIGINT	The number of disk pages that have been accessed
CREATE_PAGE	BIGINT	The number of disk pages that have been created
UNDO_READ_PAGE	BIGINT	The number of disk UNDO pages that have been read
UNDO_WRITE_PAGE	BIGINT	The number of disk UNDO pages that have been written to
UNDO_GET_PAGE	BIGINT	The number of disk UNDO pages that have been accessed
UNDO_CREATE_PAGE	BIGINT	The number of disk UNDO pages that have been created
MEM_CURSOR_FULL_SCAN	BIGINT	The number of memory table searches without indexes
MEM_CURSOR_INDEX_SCAN	BIGINT	The number of memory table searches that use indexes
DISK_CURSOR_FULL_SCAN	BIGINT	The number of disk table searches without indexes
DISK_CURSOR_INDEX_SCAN	BIGINT	The number of disk table searches that use indexes
EXECUTE_SUCCESS	BIGINT	The number of successful statement executions
EXECUTE_FAILURE	BIGINT	The number of failed statement executions
FETCH_SUCCESS	BIGINT	The number of successful fetches
FETCH_FAILURE	BIGINT	The number of failed fetches

Column name	Type	Description
PROCESS_ROW	BIGINT	The number of processed records
MEMORY_TABLE_ACCESS_COUNT	BIGINT	The number of records that a statement retrieves from the target memory table(s)
SEQNUM	INTEGER	The identifier of a wait event
EVENT	VARCHAR(128)	The name of a wait event
P1	BIGINT	Parameter 1 of the wait event
P2	BIGINT	Parameter 2 of the wait event
P3	BIGINT	Parameter 3 of the wait event
WAIT_TIME	BIGINT	The time spent waiting (in milliseconds)
SECOND_IN_TIME	BIGINT	The time spent waiting (in seconds)

Column Information

ID

This is a unique identifier that distinguishes the statement within a session.

PARENT_ID

This is the identifier of the parent statement of the given statement.

CURSOR_TYPE

A hex value of 0x02 indicates a memory cursor, whereas a hex value of 0x04 indicates a disk cursor.

SESSION_ID

This is the identifier of the session to which the statement belongs.

TX_ID

This is the identifier of the transaction that is currently being executed.

QUERY

This is a query string that is currently being executed or was executed by the statement.

LAST_QUERY_START_TIME

This is the absolute start time of execution of the most recently executed query, in seconds.

QUERY_START_TIME

This is the absolute start time of execution of the currently executed query, in seconds.

FETCH_START_TIME

If the current statement is a SELECT statement, this is the time at which the fetch started, in seconds.

EXECUTE_STATE

This is the state of the current statement. It can have the following values:

- ALLOC: The statement has been initialized and assigned.
- PREPARED: The statement has been prepared
- EXECUTED: EXECUTE of the statement has ended.
- UNKNOWN: Unkown state

FETCH_STATE

This shows the fetch state of a statement, and has the following values:

- PROCEED: FETCH in progress
- CLOSE: FETCH ended
- NO_RESULTSET: Statement that does not generate result sets
- INVALIDATED: Invalid state
- UNKNOWN: Unknown state

ARRAY_FLAG

This indicates whether or not the current statement is being executed in array or batch mode. It can have the following values:

- 0: Not executed in array or batch mode
- 1: Executed in array or batch mode

ROW_NUMBER

If the current statement is being executed in array or batch mode, this is the number of the row currently being processed, starting at 1.

EXECUTE_FLAG

Indicates whether the current statement is being executed. It can have the following values:

- 0: Not currently being executed
- 1: Currently being executed

BEGIN_FLAG

Indicates whether the current statement is being executed. It can have the following values:

- 0: Not currently being executed
- 1: Currently being executed

TOTAL_TIME

This is the total execution time of the current statement, in microseconds.

Depending on the type of the statement, the PVO time or fetch time can be added to EXECUTE_TIME.

PARSE_TIME

This is the time taken to check the syntax of the query, in microseconds.

VALIDATE_TIME

This is the time taken to validate the query, in microseconds.

OPTIMIZE_TIME

This is the time taken to optimize the query, in microseconds.

EXECUTE_TIME

This is the time actually taken to execute a query, in microseconds. In the case of a statement, this is the execution time up until the first fetch occurs.

FETCH_TIME

For a SELECT query, this is the time that elapses during fetching, in microseconds.

SOFT_PREPARE_TIME

This is the time taken to find an appropriate plan cache object in the SQL Plan Cache when creating a SQL statement and plan as part of a Prepare task. (Unit: microsecond)

SQL_CACHE_TEXT_ID

This displays the identifier of a [Parent PCO](#) or NO_SQL_CACHE_STMT.

NO_SQL_CACHE_STMT means a statement that is not registered in SQL Plan Cache. The following statements are not registered in SQL Plan Cache.

- DDL statements
- DCL statements
- Statements using NO_PLAN_CACHE hint

SQL_CACHE_PCO_ID

This is the object identifier of a [Child PCO](#).

OPTIMIZER

This is the optimization mode. It can have the following values:

- 0: Cost-based optimization
- 1: Rule-based optimization

COST

This is the cost of optimizing the query.

USED_MEMORY

Reserved for future use.

READ_PAGE

This is the number of disk data pages that are physically read when executing a query.

WRITE_PAGE

This is the number of disk data pages that are physically written to when executing a query.

GET_PAGE

This is the number of disk data pages that are accessed when executing a query.

CREATE_PAGE

This is the number of disk data pages that are created when executing a query.

UNDO_READ_PAGE

This is the number of disk UNDO pages that are physically read when executing a query.

UNDO_WRITE_PAGE

This is the number of disk UNDO pages that are physically written to when executing a query.

UNDO_CREATE_PAGE

This is the number of disk UNDO pages that are created when executing a query.

MEM_CURSOR_FULL_SCAN

This is the number of times that a memory table is searched without using an index when executing a query

MEM_CURSOR_INDEX_SCAN

This is the number of times that a memory table is searched using an index when executing a query.

DISK_CURSOR_FULL_SCAN

This is the number of times that a memory table is searched without using an index when executing a query.

DISK_CURSOR_INDEX_SCAN

This is the number of times that a memory table is searched using an index when executing a query.

EXECUTE_SUCCESS

This is the number of successful query executions.

EXECUTE_FAILURE

This is the number of failed query executions.

PROCESS_ROW

This is the number of records that were processed when a query was executed.

MEMORY_TABLE_ACCESS_COUNT

This is the total number of records that are found in memory tables when a statement is executed. It should be the same as the total number of accesses specified in the execution plan of the statement.

SEQNUM

This is the identifier of the wait event.

EVENT

This is the name of the wait event.

P1

This is a parameter used by the wait event.

P2

This is a parameter used by the wait event.

P3

This is a parameter used by the wait event.

WAIT_TIME

This is the time spent waiting (in milliseconds).

SECOND_IN_TIME

This is the time spent waiting (in seconds).

V$STATNAME

This view shows the numeric identifiers and names of statistics, and is the basis for V$SYSSTAT, which shows the overall statistics for the system, and V$SESSTAT, which shows the statistics for individual sessions.

This table alone does not have any meaning; it should be viewed through one of the above two performance views in order to provide meaningful information.

Column name	Type	Description
SEQNUM	INTEGER	The identifier for the particular statistic
NAME	VARCHAR(128)	The name of the statistic

Column Information

SEQNUM

This is the identifier of the statistic, which is shown in one of the above performance views.

NAME

This is the name of the statistic, which is shown in one of the above performance views. The serial number and a brief description of each statistic are provided in the following table. Each statistic value is expressed as a 64-bit integer in the V$SYSSTAT and V$SESSTAT performance views.

SEQ	NAME	Description
0	logon current	The number of users that are currently connected
1	logon cumulative	The cumulative number of users who have connected
2	data page read	The number of times that pages were read in the system or session
3	data page write	The number of times that pages were written to in the system or session
4	data page gets	The number of times that pages were accessed in the system or session using latches
5	data page fix	The number of times that pages were accessed in the system or session without using latches
6	data page create	The number of pages that were created in the system or session
7	undo page read	The number of times that UNDO pages were read in the system or session
8	undo page write	The number of times that UNDO pages were written to in the system or session
9	undo page gets	The number of times that UNDO pages were accessed in the system or session using latches
10	undo page fix	The number of times that UNDO pages were accessed in the system or session without using latches
11	undo page create	The number of UNDO pages that were created in the system or session

SEQ	NAME	Description
12	base time in second	The internal time that is maintained by the system (in seconds)
13	query timeout	The number of query timeouts that have occurred in the system or session
14	ddl timeout	The number of times that DDL statements have timed out in the system or session
15	idle timeout	The number of idle timeouts that have occurred in the system or session
16	fetch timeout	The number of fetch timeouts that have occurred in the system or session
17	utrans timeout	The number of utrans timeouts that have occurred in the system or session
18	session terminated	The number of sessions that have been forcibly shut down in the system
19	ddl sync timeout	The number of DDL sync timeouts that occurred in the system/session
20	statement rebuild count	The number of times that a statement has been rebuilt in the system or session
21	unique violation count	The number of times that a unique key constraint has been violated in the system or session
22	update retry count	The number of times that an update operation has been reattempted in the system or session
23	delete retry count	The number of times that a delete operation has been reattempted in the system or session
24	lock row retry count	The number of times that an attempt to lock a row has been repeated in the system or session
25	session commit	The number of commits that have occurred in the system or session
26	session rollback	The number of rollbacks that have occurred in the system or session
27	fetch success count	The number of successful fetches in the system or session
28	fetch failure count	The number of times a fetch failed in the system or session

SEQ	NAME	Description
29	execute success count	The number of times that queries were successfully executed in the system or session
30	execute success count : insert	The number of times for successfully executed select statements in the system or session.
31	execute success count : update	The number of times for successfully executed update statements in the system or session.
32	execute success count : delete	The number of times for successfully executed delete statements in the system or session.
33	execute success count : select	The number of times for successfully executed select statements in the system or session.
34	rep_execute success count : insert	The number of times for successfully executed insert statements on the replication target table.
35	rep_execute success count : update	The number of times for successfully executed update statements on the replication target table in the system/session.
36	rep_execute success count : delete	The number of time for successfully executed delete statements on the replication target table in the system/session.
37	execute failure count	The number of failures to execute a query in the system or session
38	prepare success count	The number of times that a Prepare operation was successfully conducted in the system or session
39	prepare failure count	The number of times that a Prepare operation failed in the system or session
40	rebuild count	The number of times a plan cache object was rebuilt in the system or session
41	write redo log count	The number of log records that were recorded in the system or session
42	write redo log bytes	The total number of bytes of logs that were recorded in the system or session
43	read socket count	The number of times that data were read from a socket in the system or session
44	write socket count	The number of times that data were written to a socket in the system or session

SEQ	NAME	Description
45	byte received via inet	The number of bytes of data read using an INET socket in the system or session
46	byte sent via inet	The number of bytes of data written using an INET socket in the system or session
47	byte received via unix domain	The number of bytes of data read using the Unix domain socket in the system or session
48	byte sent via unix domain	The number of bytes of data written using the Unix domain socket in the system or session
49	semop count for receiving via ipc	The number of semaphore operations for IPC read tasks in the system or session
50	semop count for sending via ipc	The number of semaphore operations for IPC write tasks in the system or session
51	memory table cursor full scan count	The number of full scan cursors (a full scan cursor is a forward-only cursor that scans an entire table) opened on memory tables using sequential read
52	memory table cursor index scan count	The number of index scan cursors opened on memory tables
53	memory table cursor GRID scan count	The number of GRID scan cursors opened on memory tables executed in the system/session.
54	disk table cursor full scan count	The number of full scan cursors opened on disk tables using sequential read
55	disk table cursor index scan count	The number of index scan cursors opened on disk tables
56	disk table cursor GRID scan count	The number of GRID cursors opened on disk tables executed in the system/session.
57	lock acquired count	The number of table locks that were obtained in the system or session (Caution: For internal reasons, when viewing V$SYSSTAT, this value may not be the same as the number of locks that have been released. However, for V$SESSTAT, the two values should be the same.)
58	lock released count	The number of table locks that have been released in the system or session

SEQ	NAME	Description
59	service thread created count	The number of service threads that have been created in the system or session
60	memory table access count	The number of times that memory tables have been accessed in the system or session
61	missing ppco x-trylatch count	The number of x-trylatch failures in Parent PCO.
62	read IB count	The number of times data was read from IB in the system/session
63	write IB count	The number of times data was written to IB by the system/session
64	byte received via IB	Data read using IB from the system/session (Unit: bytes)
65	byte sent via IB	Data written using IB in the system/session (Unit: bytes)
66	elapsed time[15]: query parse	The total amount of time taken to parse a query. This is a cumulative value.
67	elapsed time: query validate	The total amount of time taken to validate a query. This is a cumulative value.
68	elapsed time: query optimize	The total amount of time taken to optimize a query. This is a cumulative value.
69	elapsed time: query execute	The total amount of time taken to execute a query. This is a cumulative value.
70	elapsed time: query fetch	The total amount of time taken for a query to return records.
71	elapsed time: soft prepare	The total amount of time taken for soft prepare.
72	elapsed time: analyze values in DML(disk)	The total amount of time taken to analyze the input column values when executing DML statements (INSERT or UPDATE) in the system or session.
73	elapsed time: record lock validation in DML(disk)	The amount of time taken to check whether or not records can be updated in the system or session.
74	elapsed time: allocate data slot in DML(disk)	The amount of time taken to allocate data slots during a DML operation in the system or session.

SEQ	NAME	Description
75	elapsed time: write undo record in DML(disk)	The amount of time taken to write undo records in the system or session.
76	elapsed time: allocate tss in DML(disk)	The amount of time taken to allocate transaction slots in the system or session.
77	elapsed time: allocate undopage in DML(disk)	The amount of time taken to allocate undo pages in the system or session.
78	elapsed time: index operation in DML(disk)	The amount of time taken to add keys to indexes in the system or session.
79	elapsed time: create page(disk)	The amount of time taken to create pages in the system or session.
80	elapsed time: get page(disk)	The amount of time taken to access pages with latches in the system or session.
81	elapsed time: fix page(disk)	The amount of time taken to access pages without latches in the system or session.
82	elapsed time: logical aging by tx in DML(disk)	Not currently used.
83	elapsed time: physical aging by tx in DML(disk)	Not currently used.
84	elapsed time: replace (plan cache)	The time taken to replace one plan with another plan from a list.
85	elapsed time: victim free in replace (plan cache)	The time taken to release a victim while replacing one plan with another plan from a list.
86	elapsed time: hard rebuild	When a plan is found in the plan cache but is determined to be invalid, this is the amount of time taken to re-build it.
87	elapsed time: soft rebuild	When a plan is found in the plan cache but is determined to be invalid and is thus to be rebuilt, this is the amount of time spent waiting for another transaction to re-build the plan.

SEQ	NAME	Description
88	elapsed time: add hard-prepared plan to plan cache	The amount of time taken to add a plan created by hard prepare (i.e. a forcibly created plan) to the plan cache.
89	elapsed time: add hard-rebuilt plan to plan cache	The amount of time taken to add a plan created by hard rebuild (refer to #86) to the plan cache.
90	elapsed time: search time for parent PCO	The amount of time taken to find a parent PCO (Plan Cache Object that has SQL text).
91	elapsed time: creation time for parent PCO	The amount of time taken to create a new parent PCO.
92	elapsed time: search time for child PCO	The sum of #98 and #99 (i.e. 98 + 99). This is a cumulative value.
93	elapsed time: creation time for child PCO	The amount of time taken to create a new child PCO (Plan Cache Object which has an execution plan).
94	elapsed time: validation time for child PCO	The amount of time taken to validate a child PCO.
95	elapsed time: creation time for new child PCO by rebuild at execution	The amount of time taken to create a new child PCO in the case where a plan is re-built during the execution phase.
96	elapsed time: creation time for new child PCO by rebuild at soft prepare	The amount of time taken to create a new child PCO in the case where a plan is re-built during the soft prepare phase.
97	elapsed time: hard prepare time	The amount of time taken for a hard prepare, that is, to create a plan when no plan exists in the plan cache.
98	elapsed time: matching time for child PCO	The amount of time taken to determine which plan is the desired plan in the case where there are two or more child PCOs that have the same SQL text.

SEQ	NAME	Description
99	elapsed time: waiting time for hard prepare	The sum of #97 and #88 (i.e. 97 + 88). This is a cumulative value.
100	elapsed time: moving time from cold region to hot region	The amount of time taken to move a plan from a cold area to a hot area.
101	elapsed time: waiting time for parent PCO when choosing plan cache replacement victim	The amount of time spent waiting for a parent PCO latch to check child PCOs when choosing a replacement target.
102	elapsed time: privilege checking time during soft prepare	The amount of time taken to check privileges for access to objects during soft prepare.
103	elapsed time: copying logs to replication log buffer (sender side)	This is the cumulative amount of time taken for Sender Thread(s) to copy logs to the replication log buffer.
104	elapsed time: sender(s) waiting for new logs	This is the cumulative amount of time spent waiting for new logs to be written to the log buffer or log files.
105	elapsed time: sender(s) reading logs from replication log buffer	This is the cumulative amount of time that Sender Thread(s) have spent reading logs from the replication log buffer.
106	elapsed time: sender(s) reading logs from log file(s)	This is the cumulative amount of time that Sender Thread(s) have spent reading logs from log files.
107	elapsed time: sender(s) checking whether logs are useful	This is the cumulative amount of time that Sender Thread(s) have spent checking whether logs must be sent for replication.

SEQ	NAME	Description
108	elapsed time: sender(s) analyzing logs	This is the cumulative amount of time that Sender Thread(s) have spent analyzing logs and converting them into XLogs.
109	elapsed time: sender(s) sending XLogs to receiver(s)	This is the total amount of time that Sender Thread(s) have spent sending XLogs to Receiver Thread(s).
110	elapsed time: sender(s) receiving ACK from receiver(s)	This is the cumulative amount of time spent waiting for and receiving ACK from Receiver Thread(s).
111	elapsed time: sender(s) setting ACKed value	This is the total amount of time that Sender Thread(s) have spent analyzing ACK values received from Receiver Thread(s).
112	elapsed time: receiver(s) receiving XLogs from sender(s)	This is the cumulative amount of time that Receiver Thread(s) have spent receiving XLogs from Sender Thread(s).
113	elapsed time: receiver(s) performing endian conversion	This is the cumulative amount of time that Receiver Thread(s) have spent performing byte order conversion.
114	elapsed time: receiver(s) beginning transaction(s)	This is the cumulative amount of time that Receiver Thread(s) have spent beginning transactions.
115	elapsed time: receiver(s) committing transaction(s)	This is the cumulative amount of time that Receiver Thread(s) have spent committing transactions.
116	elapsed time: receiver(s) aborting transaction(s)	This is the cumulative amount of time that Receiver Thread(s) have spent rolling back transactions.
117	elapsed time: receiver(s) opening table cursor(s)	This is the cumulative amount of time that Receiver Thread(s) have spent opening table cursors.

SEQ	NAME	Description
118	elapsed time: receiver(s) closing table cursor(s)	This is the cumulative amount of time that Receiver Thread(s) have spent closing table cursors.
119	elapsed time: receiver(s) inserting rows	This is the cumulative amount of time that Receiver Thread(s) have spent inserting records.
120	elapsed time: receiver(s) updating rows	This is the cumulative amount of time that Receiver Thread(s) have spent updating records.
121	elapsed time: receiver(s) deleting rows	This is the cumulative amount of time that Receiver Thread(s) have spent deleting records.
122	elapsed time: receiver(s) opening lob cursor(s)	This is the cumulative amount of time that Receiver Thread(s) have spent opening LOB cursors.
123	elapsed time: receiver(s) preparing to write LOB(s)	This is the cumulative amount of time that Receiver Thread(s) have spent preparing to write LOBs.
124	elapsed time: receiver(s) writing LOB piece(s)	This is the cumulative amount of time that Receiver Thread(s) have spent writing LOB pieces.
125	elapsed time: receiver(s) finish writing LOBs	This is the cumulative amount of time that Receiver Thread(s) have spent finishing writing LOBs.
126	elapsed time: receiver(s) closing LOB cursor(s)	This is the cumulative amount of time that Receiver Thread(s) have spent closing LOB cursors.
127	elapsed time: receiver(s) comparing images to check for conflicts	This is the cumulative amount of time that Receiver Thread(s) have spent comparing data to check for data conflicts.
128	elapsed time: receiver(s) sending ACK	This is the cumulative amount of time that Receiver Thread(s) have spent sending ACK to Sender Thread(s).

SEQ	NAME	Description
129	elapsed time: receiver(s) trim LOB(s)	This is the accumulative time for the Receiver Threads to finish LOB trim.
130	elapsed time: task schedule	This is the total amount of accumulative time with task scheduling. (Unit: microsecond)
131	max time: task schedule	tThis is the maximum time for waiting with the task scheduling. The longest waiting time is only written.(단위: microsecond)

[15] elapsed time unit : microsecond

V$SYSSTAT

This view shows the status of the system. It should be noted that the shown value may be out of date, because the status values are updated every 3 seconds based on the data for all sessions.

Column name	Type	Description
SEQNUM	INTEGER	The identifier of the statistical category
NAME	VARCHAR(128)	The name of the statistic
VALUE	BIGINT	The value of the statistic

For information about each statistic, please refer to V$STATNAME.

Column Information

SEQNUM

This is the serial number of the system statistic.

NAME

This is the name corresponding to the statistic serial number.

VALUE

This is the current system value corresponding to the statistic serial number, expressed as a 64-bit integer.

V$SYSTEM_CONFLICT_PAGE

This displays conflict information, classified by page type, for use in analyzing bottlenecks caused by page latch contention in disk buffer space.

This information is collected only if the TIMED_STATISTICS property is set to 1.

Column name	Type	Description

Column name	Type	Description
PAGE_TYPE	VARCHAR(21)	The type of page
LATCH_MISS_CNT	BIGINT	The number of failures to acquire latches
LATCH_MISS_TIME	BIGINT	The waiting time

Column

PAGE_TYPE

This is the type of page.

LATCH_MISS_CNT

This is the number of failures to acquire buffer page latches.

LATCH_MISS_TIME

This is the amount of time (in microseconds) spent waiting for failed attempts to acquire buffer page latches.

V$SYSTEM_EVENT

This view shows cumulative statistical information about waits, classified according to wait event, from the time Altibase was started to the present.

Column name	Type	Description
EVENT	VARCHAR(128)	The name of the wait event
TOTAL_WAITS	BIGINT	The total number of waits for this event
TOTAL_TIMEOUTS	BIGINT	The number of failures to gain access to the requested resource within the specified time
TIME_WAITED	BIGINT	The total time spent waiting for this wait event by all sessions (in milliseconds)
AVERAGE_WAIT	BIGINT	The average length of a wait for this event (in milliseconds)
TIME_WAITED_MICRO	BIGINT	The total time spent waiting for this wait event by all sessions (in microseconds)
EVENT_ID	INTEGER	The identifier of the wait event
WAIT_CLASS_ID	INTEGER	The identifier of the wait class
WAIT_CLASS	VARCHAR(128)	The name of the wait class

Column Information

EVENT

This is the name of the wait event.

TOTAL_WAITS

This is the total number of waits for this event.

TOTAL_TIMEOUTS

This is the number of failures to gain access to the requested resource even after the specified time has elapsed.

TIME_WAITED

This is the total amount of time spent waiting for this wait event by all sessions (in milliseconds).

AVERAGE_WAIT

This is the average time spent waiting for this wait event (in milliseconds).

TIME_WAITED_MICRO

This is the total amount of time spent waiting for this event by all sessions (in microseconds).

EVENT_ID

This is the identifier of the wait event.

WAIT_CLASS_ID

This is the identifier of the wait class into which the event being waited for in the session is categorized.

WAIT_CLASS

This is the name of the wait class into which the event being waited for in the session is categorized.

V$SYSTEM_WAIT_CLASS

This view shows cumulative statistical information about waits, classified according to wait class, from the time Altibase was started to the present.

Column name	Type	Description
WAIT_CLASS_ID	INTEGER	The identifier of the wait class
WAIT_CLASS	VHARCHAR(128)	The name of the wait class
TOTAL_WAITS	BIGINT	The total number of waits in this wait class
TIME_WAITED	DOUBLE	The total amount of time spent waiting for this wait class by all processes (in milliseconds)

Column Information

WAIT_CLASS_ID

This is the identifier of the wait class.

WAIT_CLASS

This is the name of the wait class.

TOTAL_WAITS

This is the total number of waits for this class.

TIME_WAITED

This is the total time (in milliseconds) spent waiting for this wait class by all sessions.

Example

<Example 1> The following query outputs the waiting time and the number of waits in each wait class for all current wait events

```
iSQL> select * from v$system_wait_class order by total_waits desc;
```

<Example 2> The following query outputs the proportion of waits in each wait class to total waits and the proportion of time spent waiting in each wait class to the total amount of time spent waiting, in descending order, starting with the wait class in which the longest waits have occurred.

```
iSQL> select
            WAIT_CLASS,
            TOTAL_WAITS,
            round(100 * (TOTAL_WAITS / SUM_WAITS),2) PCT_WAITS,
            TIME_WAITED,
            round(100 * (TIME_WAITED / SUM_TIME),2) PCT_TIME
from
    (select WAIT_CLASS,
    TOTAL_WAITS,
            TIME_WAITED
     from V$SYSTEM_WAIT_CLASS
     where   WAIT_CLASS != 'Idle'),
     (select sum(TOTAL_WAITS) SUM_WAITS,
            sum(TIME_WAITED) SUM_TIME
     from V$SYSTEM_WAIT_CLASS
     where WAIT_CLASS != 'Idle')
order by 5 desc;
```

V$SYS_LICENSE_

This is a meta table that records license-related information.

Column name	Type	Description
START_DATE	VARCHAR(16)	The first inspection date using the license key
VALID_YN	VARCHAR(2)	The valid license
LICENSE_KEY	VARCHAR(128)	The corresponding license key
ISSUED_DATE	VARCHAR(16)	The date of license key issuance
TYPE	VARCHAR(32)	Standard/Enterprise/Trial
EXPIRE_DATE	VARCHAR(16)	The expire date
LICENSED_PRODUCT_VERSION	VARCHAR(8)	Major/Minor version
LICENSED_MAC_ADDRESS	VARCHAR(32)	Allowed MAC-address
LICENSED_HOST_NAME	VARCHAR(32)	Allowed hostname
LICENSED_MEM_MAX_DB_SIZE	VARCHAR(32)	The maximum memory DB size
LICENSED_DISK_MAX_DB_SIZE	VARCHAR(32)	The maximum disk DB size
LICENSED_CORE_MAX_COUNT	VARCHAR(16)	The number of cores allowed
CHECKED_MAC_ADDRESS	VARCHAR(128)	MAC-address of the current system at startup
CHECKED_HOST_NAME	VARCHAR(128)	The host name of the current system at startup
CHECKED_MEM_MAX_DB_SIZE	VARCHAR(32)	The maximum memory DB size entered when installing altibase
CHECKED_CORE_MAX_COUNT	VARCHAR(16)	The number of (physical) cores in the current system

Column Information

START_DATE

This stores the first inspection date using the license key.

VALID_YN

This indicates whether the license key is valid. If the license key is not valid, customer support may be available.

  * Y: Valid

  * N: Invalide

LICENSE_KEY

This saves the license key. If not, it is stored as -.

ISSUED_DATE

This indicates the date the license key was issued is stored.

TYPE

This stores the type of license. Standard, Enterprise, or Trial is stored. If not, it is stored as -.

EXPIRE_DATE

This stores the license expiration date. If not, it is stored as -.

LICENSED_PRODUCT_VERSIO

This stores the licensed product version in the form of MAJOR or MINOR. If not it is stored as -.

LICENSED_MAC_ADDRES

This stores the licensed hardware address. If not, it is stored as -.

LICENSED_HOST_NAME

This stores the licensed hostname. If not, it is stored as -.

LICENSED_MEM_MAX_DB_SIZE

This stores the maximum size of the licensed memory DB. The unit is bytes. If not, it is stored as -.

LICENSED_DISK_MAX_DB_SIZE

This stores the maximum size of the disk DB. The unit is bytes. If not, it is stored as -. (Not currently used)

LICENSED_CORE_MAX_COUNT

This stores the maximum number of available cores licensed. On a machine with more cores than this value, only the number of licensed cores among the total cores is used at startup.

If not, it is stored as -.

CHECKED_MAC_ADDRESS

This stores the hardware addresses currently in the system. When the nubmer is large, it is stored within the limit of 128 bytes.

CHECKED_HOST_NAME

This stores the host name of the current system.

CHECKED_MEM_MAX_DB_SIZE

This is the value of the MEM_MAX_DB_SIZE property entered by the installer when installing. The unit is bytes.

If the DB is started after changing the MEM_MAX_DB_SIZE property without starting immediately after the installation, it is saved as the changed property value.

CHECKED_MAX_CORE_COUNT

This stores the number of cores currently in the system.

Example

In order to prevent misuse, some information is processed with *.

```
iSQL> select * from system_.sys_license_;
START_DATE VALID_YN
\-------------------------------
LICENSE_KEY
\-----------------------------------------------------------------------------
--------------------------------------------------
ISSUED_DATE TYPE EXPIRE_DATE
\------------------------------------------------------------------------
LICENSED_PRODUCT_VERSION LICENSED_MAC_ADDRESS
\--------------------------------------------------------------
LICENSED_HOST_NAME LICENSED_MEM_MAX_DB_SIZE
\---------------------------------------------------------------------
LICENSED_DISK_MAX_DB_SIZE LICENSED_CORE_MAX_COUNT
\-----------------------------------------------------------
CHECKED_MAC_ADDRESS
\-----------------------------------------------------------------------------
--------------------------------------------------
CHECKED_HOST_NAME
\-----------------------------------------------------------------------------
--------------------------------------------------
CHECKED_MEM_MAX_DB_SIZE CHECKED_MAX_CORE_COUNT
\-----------------------------------------------------------
2020-5-18 Y
74BCF6CD6DE71B2341B79E4A05E327765E59C2529673CBD52CFAE779753E33A9E4FB7A746291420F3D77
5D0F0664B1EA6243CC*
2020-5-18 STANDARDEDITION 2020-10-10
7.1 -
bdw-ex-altibase 107374182400
```

```
107374182400 192
[A0:36:9F:18:DA:*][02:42:FE:4F:1A:*][52:54:00:64:F0:*][A0:36:9F:18:DA:*]
bdw-ex-altibase
10737418240 96
1 row selected.
```

V$TABLE

This view shows the list of performance views.

Column name	Type	Description
NAME	VARCHAR(39)	The name of the view
SLOTSIZE	INTEGER	The record size
COLUMNCOUNT	SMALLINT	The number of columns

Column Information

NAME

This is the name of the performance view.

SLOTSIZE

This is the size of one record in the performance view.

COLUMNCOUNT

This is the number of columns in the performance view.

V$TABLESPACES

This view shows information about tablespaces.

Column name	Type	Description
ID	INTEGER	The tablespace identifier
NAME	VARCHAR(40)	The tablespace name
NEXT_FILE_ID	INTEGER	The identifier of the next data file to be created
TYPE	INTEGER	The type of tablespace
STATE	INTEGER	The status of the tablespace
EXTENT_MANAGEMENT	VARCHAR(20)	The method of managing extents, which is set when the user creates a disk tablespace
SEGMENT_MANAGEMENT	VARCHAR(20)	The type of segment in the tablespace
DATAFILE_COUNT	INTEGER	The number of files in the tablespace

Column name	Type	Description
TOTAL_PAGE_COUNT	BIGINT	The total number of pages
EXTENT_PAGE_COUNT	INTEGER	The size of an extent (number of pages) in the tablespace
ALLOCATED_PAGE_COUNT	BIGINT	The initial number of pages in the tablespace
PAGE_SIZE	INTEGER	The size of a page in the tablespace (unit: bytes)
ATTR_LOG_COMPRESS	INTEGER	The initial number of pages in the tablespace

Column Information

ID

This is the identifier of the tablespace. The identifiers of user tablespaces start at 5 and increment.

NAME

This is the name of the tablespace, which was defined using the CREATE TABLESPACE statement.

NEXT_FILE_ID

This is an identifier that is assigned to a data file when the data file is added to the tablespace. This value increases by 1 for every individual data file that is added.

TYPE

This value indicates the type of tablespace:

- 0: MEMORY_SYSTEM_DICTIONARY
- 1: MEMORY_SYSTEM_DATA
- 2: MEMORY_USER_DATA
- 3: DISK_SYSTEM_DATA
- 4: DISK_USER_DATA
- 5: DISK_SYSTEM_TEMP
- 6: DISK_USER_TEMP
- 7: DISK_SYSTEM_UNDO
- 8: VOLATILE_USER_DATA

STATE

This value indicates the status of the tablespace.

- 1: OFFLINE
- 2: ONLINE
- 5: Offline tablespace that is being backed up
- 6: Online tablespace that is being backed up

- 128: DROPPED
- 1024: Discarded tablespace
- 1028: Discarded tablespace that is being backed up

EXTENT_MANAGEMENT

This is the method of managing extents, which is set when a user disk tablespace is created. At present, the BITMAP method is supported.

- BITMAP: This indicates whether all EXTENTs of a tablespace are allocated.

SEGMENT_MANAGEMENT

When a segment is created in a tablespace, this indicates which type of segment is to be created.

- MANUAL: This indicates that a Free list Management Segment (FMS) is to be created.
- AUTO: This indicates that a bitmap-based Tree Management Segment (TMS) is to be created.

DATAFILE_COUNT

This is the number of data files in the tablespace.

TOTAL_PAGE_COUNT

This is the size of the tablespace, expressed as the number of pages. The actual size of the tablespace can be calculated by multiplying this value by the page size (TOTAL_PAGE_COUNT * PAGE_SIZE). This is the actual number of usable pages, and does not include the single file header page for each file.

EXTENT_PAGE_COUNT

This is the size of an extent for this tablespace, expressed as the number of pages. An extent has at least 3 pages.

ALLOCATED_PAGE_COUNT

This is the initial number of pages that were allocated to the tablespace.

PAGE_SIZE

This is the size of each of the pages in the tablespace. It is 8 kB for disk tablespaces and 32 kB for memory tablespaces.

ATTR_LOG_COMPRESS

This indicates whether to perform log compression when executing DML statements on tables in the tablespace.

- 0: do not compress logs
- 1: compress logs

V$TIME_ZONE_NAMES

This view displays region names, abbreviations and UTC offset values available to be set for the TIME_ZONE property.

Column name	Type	Description
NAME	VARCHAR(40)	The region name or abbreviation
UTC_OFFSET	VARCHAR(6)	The UTC offset

Column Information

NAME

This is the character string or abbreviation of a region name used to set the time zone such as Asia/Seoul or KST.

UTC_OFFSET

This is the offset value from the UTC(Coordinated Universal Time) of the time zone. For example, the UTC offset value for Asia/Seoul is +09:00.

V$TRACELOG

This view displays information related to message logging, for use in leaving records related to internal database operation.

Column name	Type	Description
MODULE_NAME	VARCHAR(16)	The name of the module
TRCLEVEL	INTEGER	The logging level (1~32)
FLAG	VARCHAR(8)	Whether logging is enabled for this module and level.
POWLEVEL	BIGINT	Two to the power of the level minus one (2^(TRCLEVEL-1))
DESCRIPTION	VARCHAR(64)	A description of this module and level

Column Information

MODULE_NAME

This is the name of an Altibase module. At present, Altibase consists of the SERVER, QP, RP and SM modules, each of which can perform message logging.

TRCLEVEL

This is the message logging level. It has a value between 1 and 32.

FLAG

This displays the setting that determines whether history messages for this module and level are output.

- X: Not output

- O: Output

- SUM: This value indicates that the POWLEVEL column for this record contains the sum of POWLEVELs for which the FLAG is set to 'O' in each module

For information about output settings, please refer to the following description.

POWLEVEL

This is 2 to the power of the TRCLEVEL minus one, that is, $2^{(TRCLEVEL-1)}$. The stored procedures addTrcLevel() and delTrcLevel() are provided so that users can easily set the logging level. These stored procedures can be created by executing tracelog.sql, which comes with the package.

DESCRIPTION

This is an explanation of the corresponding module and level.

Example

To check the trace logging level currently set for the server module:

```
iSQL> select module_name, trclevel, flag, powlevel, description from v$tracelog
where module_name like '%SER%';
MODULE_NAME TRCLEVEL FLAG POWLEVEL DESCRIPTION
------------------------------------------------
SERVER 1 O 1 [DEFAULT] TimeOut(Query,Fetch,Idle,UTrans) Trace Log
SERVER 2 O 2 [DEFAULT] Network Operation Fail Trace Log
SERVER 3 O 4 [DEFAULT] Memory Operation Warning Trace Log
SERVER 4 X 8 ---
SERVER 5 X 16 ---
SERVER 6 X 32 ---
SERVER 7 X 64 ---
SERVER 8 X 128 ---
SERVER 9 X 256 ---
SERVER 10 X 512 ---
SERVER 11 X 1024 ---
SERVER 12 X 2048 ---
SERVER 13 X 4096 ---
SERVER 14 X 8192 ---
SERVER 15 X 16384 ---
SERVER 16 X 32768 ---
SERVER 17 X 65536 ---
SERVER 18 X 131072 ---
SERVER 19 X 262144 ---
```

```
SERVER 20 X 524288 ---
SERVER 21 X 1048576 ---
SERVER 22 X 2097152 ---
SERVER 23 X 4194304 ---
SERVER 24 X 8388608 ---
SERVER 25 X 16777216 ---
SERVER 26 X 33554432 ---
SERVER 27 X 67108864 ---
SERVER 28 X 134217728 ---
SERVER 29 X 268435456 ---
SERVER 30 X 536870912 ---
SERVER 31 X 1073741824 ---
SERVER 32 X 2147483648 ---
SERVER 99 SUM 7 Total Sum of Trace Log Values
33 rows selected.
```

Usage

Altibase provides 6 messages logging properties such as the SERVER, SM, QP, RP, RP_CONFILICT, DR as follows.

- SERVER_MSGLOG_FLAG: Communication and server messages

- SM _MSGLOG_FLAG: Storage manager-related messages

- QP_MSGLOG_FLAG:: Query processor-related messages

- RP_MSGLOG_FLAG: Replication-related messages

- RP_CONFLICT_MSGLOG_FLAG: Replication conflict-related message

- LB_MSGLOG_FLAG: Service thread action-related messages

Each property can be set to 32bits. Refer to V$TRACELOG for more information on each message type and details.

The message logging details can be changed as follows.

- To disable the output of all server logging messages:

```
alter system set server_msglog_flag=0
```

- To enable the output of server logging messages related to the 1st, 2nd and 5th bits (1+2+5):

```
alter system set server_msglog_lfag=8
```

- To disable the output of all replication logging messages except conflict-related messages:

```
alter system set rp_msglog_flag=2
```

- To enable stored procedure error line logging (the 1st bit) and details pertaining to the execution of DDL statements (the 2nd bit) for the query processor (1+2):

```
alter system set qp_msglog_flag=3
```

- To disable the output of all replication conflict-related logging messages except SQL(the 3rd bit):

```
alter system set rp_conflict_msglog_flag=4
```

V$TRANSACTION

This view displays information about transaction objects.

Column name	Type	Description
ID	BIGINT	The transaction identifier
SESSION_ID	INTEGER	See below
MEMORY_VIEW_SCN	VARCHAR(29)	See below
MIN_MEMORY_LOB_VIEW_SCN	VARCHAR(29)	See below
DISK_VIEW_SCN	VARCHAR(29)	See below
MIN_DISK_LOB_VIEW_SCN	VARCHAR(29)	See below
COMMIT_SCN	VARCHAR(29)	See below
STATUS	BIGINT	See below
UPDATE_STATUS	BIGINT	See below
LOG_TYPE	INTEGER	See below
XA_COMMIT_STATUS	BIGINT	See below
XA_PREPARED_TIME	VARCHAR(64)	See below
FIRST_UNDO_NEXT_LSN_LFGID	INTEGER	Not used (0)
FIRST_UNDO_NEXT_LSN_FILENO	INTEGER	See below
FIRST_UNDO_NEXT_LSN_OFFSET	INTEGER	See below
CURRENT_UNDO_NEXT_SN	BIGINT	For internal use
CURRENT_UNDO_NEXT_LSN_LFGID	INTEGER	Not used (0)
CURRENT_UNDO_NEXT_LSN_FILENO	INTEGER	For internal use
CURRENT_UNDO_NEXT_LSN_OFFSET	INTEGER	For internal use
LAST_UNDO_NEXT_LSN_LFGID	INTEGER	Not used (0)
LAST_UNDO_NEXT_LSN_FILENO	INTEGER	See below

Column name	Type	Description
LAST_UNDO_NEXT_LSN_OFFSET	INTEGER	See below
LAST_UNDO_NEXT_SN	BIGINT	See below
SLOT_NO	INTEGER	See below
UPDATE_SIZE	BIGINT	See below
ENABLE_ROLLBACK	BIGINT	For internal use
FIRST_UPDATE_TIME	INTEGER	See below
LOG_BUF_SIZE	INTEGER	For internal use
LOG_OFFSET	INTEGER	For internal use
SKIP_CHECK_FLAG	BIGINT	For internal use
SKIP_CHECK_SCN_FLAG	BIGINT	For internal use
DDL_FLAG	BIGINT	See below
TSS_RID	BIGINT	See below
RESOURCE_GROUP_ID	INTEGER	The log file group identifier
LEGACY_TRANS_COUNT	INTEGER	For internal use
ISOLATION_LEVEL	INTEGER	See below

Column Information

ID

This is a number for classifying the transaction, ranging from 0 to 2^32 – 1. These values can be reused.

SESSION_ID

This is the identifier of the session in which the transaction is executing. If no session is associated with the transaction, this value is -1, which indicates that the transaction branch is in a prepared state in an XA environment.

MEMORY_VIEW_SCN

Because Altibase uses MVCC, it has an SCN that indicates the relative point in time at which each cursor for a table was opened. This value is the smallest value of the View SCNs for memory table cursors for the transaction. A value of 2^63 means that no cursor is open.

MIN_MEMORY_LOB_VIEW_SCN

This is the SCN of the oldest of the currently open disk LOB cursors for the present transaction. A value of 2^63 means that no cursors are open.

DISK_VIEW_SCN

This is the lowest of the View SCN values for cursors that are currently open for disk tables for the present transaction. The range of values is the same as for MEMORY_VIEW_SCN.

MIN_DISK_LOB_VIEW_SCN

This is the SCN of the oldest of the currently open disk LOB cursors for the present transaction. A value of 2^63 means that no cursors are open.

COMMIT_SCN

This is the system SCN at the point in time at which the transaction is committed. A value of 263 means that the transaction has not been committed yet.

STATUS

This is the status of the current transaction. The possible values are:

- 0: BEGIN
- 1: PRECOMMIT
- 2: COMMIT_IN_MEMORY
- 3: COMMIT
- 4: ABORT
- 5: BLOCKED
- 6: END

UPDATE_STATUS

This indicates whether the transaction is a transaction that is still updating or a read-only transaction.

- 0: read-only
- 1: updating

LOG_TYPE

This indicates whether the transaction updates tables related to replication. The possible values are:

- 0: General
- 1: Replication-related

XA_COMMIT_STATUS

This is the status of a local transaction that is caused by a global transaction. It can have the following values:

- 0: BEGIN

- 1: PREPARED
- 2: COMPLETE

XA_PREPARED_TIME

This is the point in time at which a PREPARE command was received from the global transaction manager as the result of a global transaction.

FIRST_UNDO_NEXT_LSN_FILENO

This is the file number portion of the LSN, which indicates the location of the first log recorded for the transaction.

FIRST_UNDO_NEXT_LSN_OFFSET

This is the offset portion of the LSN, which indicates the location of the first log recorded for the transaction. The offset indicates the location of the log within a file.

LAST_UNDO_NEXT_LSN_FILENO

This is the file number portion of the LSN, which indicates the location of the last log recorded for the transaction.

LAST_UNDO_NEXT_LSN_OFFSET

This is the offset portion of the LSN, which indicates the location of the last log recorded for the transaction. The offset indicates the location of the log within a file.

LAST_UNDO_NEXT_SN

This is the sequence number (SN) of the last log recorded for the transaction.

SLOT_NO

This is the location of the transaction object in the transaction pool.

UPDATE_SIZE

This is the size of the data created as the result of an UPDATE operation executed by the transaction. If this value is greater than the value of the LOCK_ESCALATION_MEMORY_SIZE property, the table is locked with an X-lock and updates are performed according to the in-place update method.

FIRST_UPDATE_TIME

This is the point in time at which the database was first updated.

DDL_FLAG

This indicates whether the transaction is one that executes a DLL statement:

- 0: non-DDL
- 1: DDL

TSS_RID

This is the physical location of the Transaction Status Slot (TSL), which is obtained in order to perform an UPDATE operation on a disk table. A nonzero value means that the transaction has executed at least one update operation on a disk table.

ISOLATION_LEVEL

This is the isolation level of transaction.

- 0: READ COMMITTED
- 1: REPEATABLE READ
- 2: SERIALIZABLE

V$TRANSACTION_MGR

This value displays information about the Altibase Transaction Manager.

Column name	Type	Description
TOTAL_COUNT	INTEGER	The total number of transactions
FREE_LIST_COUNT	INTEGER	The number of free lists
BEGIN_ENABLE	BIGINT	Indicates whether a new transaction can be commenced
ACTIVE_COUNT	INTEGER	The number of active transactions
SYS_MIN_DISK_VIEWSCN	VARCHAR(29)	The lowest transaction disk view SCN

Column Information

TOTAL_COUNT

When Altibase is started, it creates a number of transaction objects equal to the number defined in this property, and uses these objects as the transaction pool. TOTAL_COUNT is the total number of transactions that have been created.

FREE_LIST_COUNT

This is the number of lists used to separately manage the transaction pool.

BEGIN_ENABLE

This indicates whether a new transaction can begin.

- 0: disabled
- 1: enabled

ACTIVE_COUNT

This is the number of transaction objects that have been assigned to tasks and are currently executing them.

SYS_MIN_DISK_VIEWSCN

This is the lowest transaction disk view SCN (System Change Number).

V$TSSEGS

This view outputs a list of all TSS segments that exist in the undo tablespace.

Column name	Type	Description
SPACE_ID	INTEGER	The identifier of the undo tablespace
SEG_PID	INTEGER	The identifier of the TSS segment page
TXSEG_ENTRY_ID	INTEGER	The identifier of the transaction segment
CUR_ALLOC_EXTENT_RID	BIGINT	The RID of the extent currently being used in the TSS segment
CUR_ALLOC_PAGE_ID	INTEGER	The identifier of the page currently being used in the TSS segment
TOTAL_EXTENT_COUNT	BIGINT	The total number of extents in the TSS segment
TOTAL_EXTDIR_COUNT	BIGINT	The total number of extent directories in the TSS segment
PAGE_COUNT_IN_EXTENT	INTEGER	The total number of pages in one extent

Column Information

SPACE_ID

This is the identifier of the undo tablespace.

SEG_PID

This is the identifier of the TSS segment page.

TXSEG_ENTRY_ID

This is the identifier of the transaction segment.

CUR_ALLOC_EXTENT_RID

This is the RID (resource identifier) of the extent currently being used in the TSS segment.

CUR_ALLOC_PAGE_ID

This is the identifier of the page currently being used in the TSS segment.

TOTAL_EXTENT_COUNT

This is the total number of extents in the TSS segment.

TOTAL_EXTDIR_COUNT

This is the total number of extent directories in the TSS segment.

PAGE_COUNT_IN_EXTENT

This is the total number of pages in one extent.

V$TXSEGS

This view outputs the list of transaction segments that are bound to transactions, and thus online (active).

Column name	Type	Description
ID	INTEGER	The identifier of the transaction segment
TRANS_ID	BIGINT	The identifier of the transaction to which the segment is bound
MIN_DISK_VIEW_SCN	VARCHAR(29)	The lowest disk view SCN of the transaction
COMMIT_SCN	VARCHAR(29)	The commit SCN of the transaction
FIRST_DISK_VIEW_SCN	VARCHAR(29)	The first disk view SCN of the transaction
TSS_RID	BIGINT	The RID of the TSS for the transaction
TSSEG_EXTENT_RID	BIGINT	The RID of the extent of the TSS segment allocated to the TSS
FST_UDSEG_EXTENT_RID	BIGINT	The RID of the first extent of the UNDO segment used by the transaction
LST_UDSEG_EXTENT_RID	BIGINT	The RID of the last extent of the UNDO segment used by the transaction
FST_UNDO_PAGEID	INTEGER	The identifier of the page containing the first UNDO record written by the transaction
FST_UNDO_SLOTNUM	SMALLINT	The slot number of the first UNDO record written by the transaction
LST_UNDO_PAGEID	INTEGER	The identifier of the page containing the last UNDO record written by the transaction

Column name	Type	Description
LST_UNDO_SLOTNUM	SMALLINT	The slot number of the last UNDO record written by the transaction

Column Information

ID

This is the identifier of the transaction segment.

TRANS_ID

This is the identifier of the transaction to which the segment is bound.

MIN_DISK_VIEW_SCN

This is the lowest disk view SCN for the transaction.

COMMIT_SCN

This is the commit SCN for the transaction.

FIRST_DISK_VIEW_SCN

This is the first disk view SCN for the transaction.

TSS_RID

This is the RID (resource identifier) of the TSS (Transaction Status Slot) allocated to the transaction.

TSSEG_EXTENT_RID

This is the RID (resource identifier) of the extent of the TSS segment allocated to the TSS.

FST_UDSEG_EXTENT_RID

This is the RID (resource identifier) of the first extent of the UNDO segment used by the transaction.

LST_UDSEG_EXTENT_RID

This is the RID (resource identifier) of the last extent of the UNDO segment used by the transaction.

FST_UNDO_PAGEID

This is the identifier of the page containing the first UNDO record written when the transaction is updated.

FST_UNDO_SLOTNUM

This is the slot number in the page containing the first UNDO record written when the transaction is updated.

LST_UNDO_PAGEID

This is the identifier of the page containing the last UNDO record written when the transaction is updated.

LST_UNDO_SLOTNUM

This is the slot number in the page containing the last UNDO record written when the transaction is updated.

V$UDSEGS

This view outputs a list of all UNDO segments existing in the undo tablespace.

Column name	Type	Description
SPACE_ID	INTEGER	The undo tablespace identifier
SEG_PID	INTEGER	The UNDO segment page identifier
TXSEG_ENTRY_ID	INTEGER	The transaction segment identifier
CUR_ALLOC_EXTENT_RID	BIGINT	The RID of the extent currently used in the UNDO segment
CUR_ALLOC_PAGE_ID	INTEGER	The identifier of the page currently used in the UNDO segment
TOTAL_EXTENT_COUNT	BIGINT	The total number of extents in the UNDO segment
TOTAL_EXTDIR_COUNT	BIGINT	The total number of extent directories in the UNDO segment
PAGE_COUNT_IN_EXTENT	INTEGER	The total number of pages in one extent

Column Information

SPACE_ID

This is the identifier of the undo tablespace.

SEG_PID

This is the identifier of the page associated with the UNDO segment.

TXSEG_ENTRY_ID

This is the identifier of the segment used by the transaction.

CUR_ALLOC_EXTENT_RID

This is the RID of the extent that is currently being used in the UNDO segment.

CUR_ALLOC_PAGE_ID

This is the identifier of the page that is currently being used in the UNDO segment.

TOTAL_EXTENT_COUNT

This is the total number of extents in the UNDO segment.

TOTAL_EXTDIR_COUNT

This is the total number of extent directories in the UNDO segment.

PAGE_COUNT_IN_EXTENT

This is the total number of pages in one extent.

V$UNDO_BUFF_STAT

This view displays buffer pool statistics related to the undo tablespace.

Column name	Type	Description
READ_PAGE_COUNT	BIGINT	See below
GET_PAGE_COUNT	BIGINT	The number of page requests made to the buffer manager
FIX_PAGE_COUNT	BIGINT	The number of UNDO page requests made to the buffer manager
CREATE_PAGE_COUNT	BIGINT	See below
HIT_RATIO	DOUBLE	The hit ratio of the buffer frame

Column Information

READ_PAGE_COUNT

The total number of pages read from disk since the buffer was initialized.

GET_PAGE_COUNT

This is the total number of page requests made to the buffer manager since the buffer was initialized. If the page is in the buffer, the buffer manager returns the requested page, otherwise the page is read from disk and then returned.

FIX_PAGE_COUNT

This is the total number of UNDO page requests made without latches to the buffer manager since the buffer was initialized.

CREATE_PAGE_COUNT

This is the total number of page creation requests made by transactions to the buffer manager since the buffer was initialized. The buffer manager responds to such requests by obtaining a free BCB from the buffer and then creating and returning a page. This operation does not incur any disk I/O.

V$USAGE

This view outputs information about the amount of space used by all of the tables and indexes that exist in the database. In order for the information presented in this view to be correct, it is first necessary to execute the built-in DBMS Stat stored procedures to gather statistical information.

For a detailed explanation of the built-in DBMS Stat stored procedures, please refer to the *Stored Procedures Manual*.

Column name	Type	Description
TYPE	CHAR(1)	The type of the object
TARGET_ID	BIGINT	An identifier for the object
META_SPACE	BIGINT	The amount of space occupied by meta information about the object
USED_SPACE	BIGINT	The amount of space occupied by the actual data in the object
AGEABLE_SPACE	BIGINT	The amount of space occupied by outdated data that must be retained for concurrency control
FREE_SPACE	BIGINT	The amount of free space in the object

Column Information

TYPE

This indicates the type of object. The value is "T" for a table and "I" for an index.

TARGET_ID

This is an identifier for the object. For a table, it is TABLE_OID (the table object identifier), whereas for an index it is INDEX_ID. To output the name of the object, use this column to join this table to the SYSTEM.*SYS_TABLES* meta table using the TABLE_OID column, or to the SYSTEM.*SYS_INDICES* meta table using the INDEX_ID column.

META_SPACE

This is the amount of space used to store the meta information for the object.

USED_SPACE

This is the amount of space used to store the actual data contained by the object.

AGEABLE_SPACE

Because MVCC is implemented in Altibase, even after data has already been deleted from a table or an index, previous versions of data are maintained for a short time in order to support concurrency control. This column indicates the amount of space occupied by such data.

FREE_SPACE

This is the amount of space in the object that has either never been used, or that was used but has since been freed and can be reused.

Example

```
iSQL> exec gather_database_stats();
SYSTEM_.SYS_TABLES_
SYSTEM_.SYS_COLUMNS_
SYSTEM_.SYS_DATABASE_
SYSTEM_.SYS_USERS_
SYSTEM_.SYS_DN_USERS_
SYSTEM_.SYS_TBS_USERS_
SYSTEM_.SYS_INDICES_
SYSTEM_.SYS_INDEX_COLUMNS_
...
Execute success.

iSQL> DESC V$USAGE;
[ ATTRIBUTE ]
--------------------------------------------------------------------------
NAME                                    TYPE
--------------------------------------------------------------------------
TYPE                                    CHAR(1)
TARGET_ID                               BIGINT
META_SPACE                              BIGINT
USED_SPACE                              BIGINT
AGABLE_SPACE                            BIGINT
FREE_SPACE                              BIGINT

iSQL> select * from v$usage limit 10;
V$USAGE.TYPE  V$USAGE.TARGET_ID    V$USAGE.META_SPACE   V$USAGE.USED_SPACE
V$USAGE.AGABLE_SPACE V$USAGE.FREE_SPACE
--------------------------------------------------------------------------------
-----------------------------------------
T   65568               128                 12672                0
    19968
```

```
I   5                   0                   528                 0
    1520
I   6                   0                   528                 0
    1520
I   7                   0                   528                 0
    1520
I   8                   0                   528                 0
    1520
T   67976               464                 66624               0
    63984
I   9                   0                   3240                0
    856
I   10                  0                   3240                0
    856
I   11                  0                   3240                0
    856
T   89648               848                 2128                0
    29792
10 rows selected.
```

V$VERSION

This view displays information about the version of the database.

Column name	Type	Description
PRODUCT_VERSION	VARCHAR(128)	The product version, e.g. 6.1.1.1
PKG_BUILD_PLATFORM_INFO	VARCHAR(128)	The platform on which the package was built
PRODUCT_TIME	VARCHAR(128)	The date on which the package was built
SM_VERSION	VARCHAR(128)	The version of the Storage Manager
META_VERSION	VARCHAR(128)	The meta table version
PROTOCOL_VERSION	VARCHAR(128)	The communication protocol version
REPL_PROTOCOL_VERSION	VARCHAR(128)	The replication protocol version

Column Information

PRODUCT_VERSION

This is the version of the Altibase product.

PKG_BUILD_PLATFORM_INFO

This is information about the platform on which the package was built.

PRODUCT_TIME

This is the date and time when the current package was built on the platform.

SM_VERSION

This is the version of the Storage Manager. This version information changes every time the storage structure changes.

META_VERSION

This is the version of the meta tables, in which database information is managed.

PROTOCOL_VERSION

This is the version of the protocols used for database communication.

REPL_PROTOCOL_VERSION

This is the version of the protocol used for replication.

V$VOL_TABLESPACES

This view shows information about volatile tablespaces, which exist in memory.

Column name	Type	Description
SPACE_ID	INTEGER	The identifier of the tablespace
SPACE_NAME	VARCHAR(512)	The name of the tablespace
SPACE_STATUS	INTEGER	The status of the tablespace
INIT_SIZE	BIGINT	The initial size of the tablespace (in bytes)
AUTOEXTEND_MODE	INTEGER	The auto extension mode of the tablespace
NEXT_SIZE	BIGINT	The auto extension increment size (in bytes)
MAX_SIZE	BIGINT	The maximum size of the tablespace (in bytes)
CURRENT_SIZE	BIGINT	The current size of the tablespace (in bytes)
ALLOC_PAGE_COUNT	BIGINT	The total number of pages in the tablespace
FREE_PAGE_COUNT	BIGINT	The number of free pages in the tablespace

Column Information

SPACE_STATUS

This is a value indicating the status of the tablespace. Please refer to V$MEM_TABLESPACE_STATUS_DESC for details.

AUTOEXTEND_MODE

This indicates the Autoextend mode. If it is set to 1, Autoextend is enabled; if not, Autoextend is disabled.

NEXTSIZE

This is the size to expand during auto expansion (in bytes).

MAXSIZE

This is the maximum size of the tablespace (in bytes).

CURRENT_SIZE

This is the current size of the tablespace (in bytes).

ALLOC_PAGE_COUNT

This is the number of pages in the tablespace.

FREE_PAGE_COUNT

This is the number of free pages in the tablespace.

V$WAIT_CLASS_NAME

This view shows information for classifying Altibase server wait events. This performance view can be used to check wait classes, which are a higher concept for classifying the various kinds of wait events.

Column name	Type	Description
WAIT_CLASS_ID	INTEGER	The identifier of the wait class
WAIT_CLASS	VARCHAR(128)	The name of the wait class

Column Information

WAIT_CLASS_ID

This is the class identifier of the wait event.

WAIT_CLASS

This is the wait class, which is a higher concept for classifying and grouping wait events. In Altibase, wait events are classified into the following 8 wait event classes:

WAIT_CLASS_ID	WAIT_CLASS	Description

WAIT_CLASS_ID	WAIT_CLASS	Description
0	Other	This wait class includes all wait events not included in any of the following classes.
1	Administrative	This class includes wait events that make the user wait due to the execution of a command with SYSDBA privileges.
2	Configuration	This class includes wait events pertaining to unsuitable settings for database resources.
3	Concurrency	This class includes wait events pertaining to internal database resources.
4	Commit	This class includes wait events pertaining to the synchronization of REDO logs in log files
5	Idle	This class includes wait events pertaining to requested tasks in sessions.
6	User I/O	This class includes wait events pertaining to user I/O.
7	System I/O	This class includes wait events pertaining to system I/O.
8	Replication	This class includes wait events pertaining to replication.

V$XID

This view displays a list of XIDs, which are identifiers for distributed transactions in the DBMS. In compliance with XA, the distributed transaction identifier is generated internally by the TM (Transaction Manager) and sent to the RM (Resource Manager), that is, to other database nodes, when a distributed transaction commences.

Column name	Type	Description
XID_VALUE	VARCHAR(256)	This returns the XID value as a character string
ASSOC_SESSION_ID	INTEGER	The identifier of the session connected to the XID object
TRANS_ID	INTEGER	The identifier of the distributed transaction within the XID object
STATE	VARCHAR(24)	The state of the XID object
STATE_START_TIME	INTEGER	The time at which the state of the XID object was determined
STATE_DURATION	BIGINT	The amount of time that has elapsed since the state of the XID was determined

Column name	Type	Description
TX_BEGIN_FLAG	VARCHAR(9)	A flag within the XID object indicating whether the transaction has begun
REF_COUNT	INTEGER	The number of current references to the XID object

Column Information

XID_VALUE

This is the XID value, expressed as a character string.

ASSOC_SESSION_ID

This is the identifier of the session related to the XID object, that is, the session which executed XA_START for this XID

TRANS_ID

This is the internal identifier of the distributed transaction within the XID object.

STATE

This is the state of execution of the XID object. The possible values for this state are as follows:

- IDLE: This means that no sessions are connected to the XID.
- ACTIVE: This means that there is a session connected to the XID. In other words, XA_START has been executed for this XID.
- PREPARED: This means that a Prepare command has been received for a 2PC (Phase Commit) task.
- HEURISTICALLY_COMMITED: This means that the DBMS has forcefully committed the transaction branch of the XID.
- HEURISTICALLY_ROLLBACKED: This means that the DBMS has forcefully rolled back the transaction branch of the XID.
- NO_TX: This means that the XID has just been initialized, or that the transaction branch related to the XID has been committed or rolled back.

STATE_START_TIME

This is the time at which the state of the XID object was determined.

STATE_DURATION

This is the amount of time that has elapsed since the state of the XID object was determined.

TX_BEGIN_FLAG

This is an internal flag within the XID object that indicates whether the transaction branch has been started in the RM.

- BEGIN: The transaction has started

- NOT BEGIN: The transaction has not started

REF_COUNT

This is the number of current references to the XID object.

2. Sample Schema

This appendix provides information about the schemas and data used in the examples in the Altibase Manuals.

Information about the Sample Schema

Script Files

A schema creation file is provided at $ALTIBASE_HOME/sample/APRE/schema/schema.sql.

Executing this file creates the tables referenced in the manuals and populates them with sample data.

Therefore, if you would like to work with the examples described in the manuals, first execute the schema creation file, after which it will be possible to follow the provided examples.

The Sample Schema

Purpose: Managing Customers and Orders

Tables: employees, departments, customers, orders, goods

employees Table

Primary Key: Employee Number (eno)

Column Name	Data Type	Description	Other
eno	INTEGER	Employee Number	PRIMARY KEY
e_lastname	CHAR(20)	Employee Last Name	NOT NULL
e_firstname	CHAR(20)	Employee First Name	NOT NULL
emp_job	VARCHAR(15)	Title	NULL allowed
emp_tel	CHAR(15)	Telephone Number	NULL allowed
dno	SMALLINT	Department Number	NULL allowed, INDEX ASC
salary	NUMBER(10,2)	Monthly Salary	NULL allowed, DEFAULT 0
sex	CHAR(1)	Sex (Gender)	NULL allowed
birth	CHAR(6)	Birthday	NULL allowed
join_date	DATE	Date of entry	NULL allowed
status	CHAR(1)	Status	NULL allowed, DEFAULT 'H'

departments Table

Primary Key: Department Number (dno)

Column Name	Data Type	Description	Other
dno	SMALLINT	Department Number	PRIMARY KEY
dname	CHAR(30)	Department Name	NOT NULL
dep_location	CHAR(15)	Department Location	NULL allowed
mgr_no	INTEGER	Administrator Number	NULL allowed, INDEX ASC

customers Table

Primary Key: Resident Registration Number (cno)

Column Name	Data Type	Description	Other
cno	CHAR(14)	Customer Number	PRIMARY KEY
c_lastname	CHAR(20)	Customer Last Name	NOT NULL
c_firstname	CHAR(20)	Customer First Name	NOT NULL
cus_job	VARCHAR(20)	Occupation	NULL allowed
cus_tel	NIBBLE(15)	Telephone Number	NOT NULL
sex	CHAR(1)	Sex (Gender)	NOT NULL
birth	CHAR(6)	Birthday	NULL allowed
postal_cd	VARCHAR(9)	Postal Code	NULL allowed
address	VARCHAR(60)	Address	NULL allowed

orders Table

Primary Keys: Order Number & Order Date (ono, order_date)

Column Name	Data Type	Description	Other
ono	BIGINT	Order Number	PRIMARY KEY
order_date	DATE	Order Date	PRIMARY KEY
eno	INTEGER	Sales Clerk	NOT NULL, INDEX ASC
cno	BIGINT	Customer Number	NOT NULL, INDEX DESC

Column Name	Data Type	Description	Other
gno	CHAR(10)	Product No.	NOT NULL, INDEX ASC
qty	INTEGER	Order Quantity	NULL allowed, DEFAULT 1
arrival_date	DATE	Expected Arrival Date	NULL allowed
processing	CHAR(1)	Order Status	NUL allowed L, O: ORDER, R: PREPARE, D: DELIVERY, C: COMPLETE, DEFALT 'O'

goods Table

Primary Key: Product No. (gno)

Column Name	Data Type	Description	Other
gno	CHAR(10)	Product Number	PRIMARY KEY
gname	CHAR(20)	Product Name	NOT NULL, UNIQUE
goods_location	CHAR(9)	Storage Location	NULL allowed
stock	INTEGER	Stored Quantity	NULL allowed, DEFAULT 0
price	NUMERIC(10,2)	Item Price	NULL allowed

dual Table

Record Size: 1

Column Name	Data Type	Description	Other
DUMMY	CHAR(1)		

E-R Entity-Relationship (ER) Diagram and Sample Data

E-R Diagram
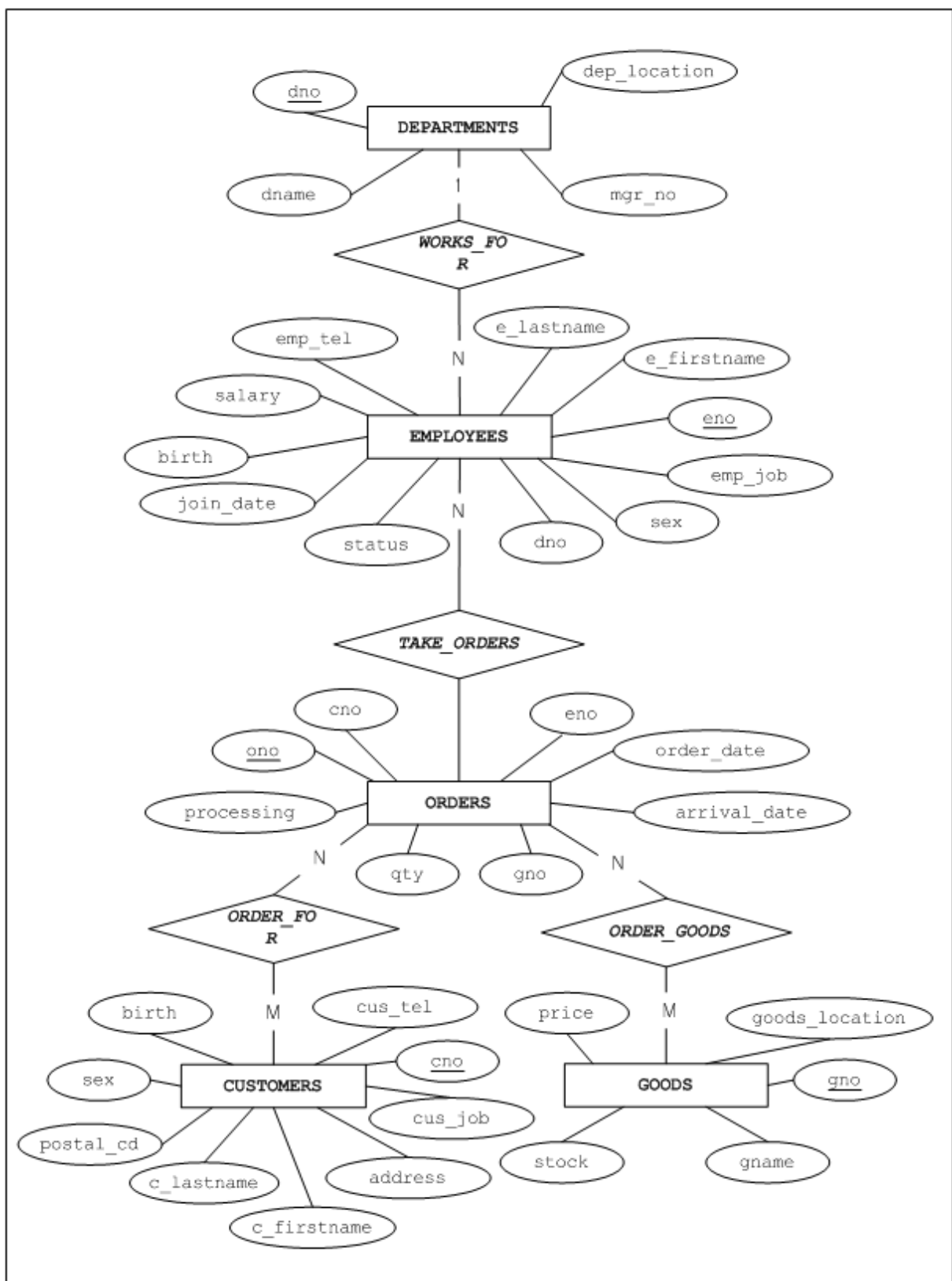
Sample Data

Employees Table

```
iSQL> select * from employees;
ENO          E_LASTNAME              E_FIRSTNAME             EMP_JOB
--------------------------------------------------------------------------
EMP_TEL          DNO         SALARY     SEX  BIRTH   JOIN_DATE     STATUS
--------------------------------------------------------------------------
1            Moon                    Chan-seung              CEO
01195662365      3002                    M                               R
2            Davenport               Susan                   designer
0113654540                   1500       F   721219  18-NOV-2009  H
3            Kobain                  Ken                     engineer
0162581369       1001        2000       M   650226  11-JAN-2010  H
4            Foster                  Aaron                   PL
0182563984       3001        1800       M   820730               H
5            Ghorbani                Farhad                  PL
01145582310      3002        2500       M           20-DEC-2009  H
6            Momoi                   Ryu                     programmer
0197853222       1002        1700       M   790822  09-SEP-2010  H
7            Fleischer               Gottlieb                manager
0175221002       4002        500        M   840417  24-JAN-2004  H
8            Wang                    Xiong                   manager
0178829663       4001                   M   810726  29-NOV-2009  H
9            Diaz                    Curtis                  planner
0165293668       4001        1200       M   660102  14-JUN-2010  H
10           Bae                     Elizabeth               programmer
0167452000       1003        4000       F   710213  05-JAN-2010  H
11           Liu                     Zhen                    webmaster
0114553206       1003        2750       M           28-APR-2011  H
12           Hammond                 Sandra                  sales rep
0174562330       4002        1890       F   810211  14-DEC-2009  H
13           Jones                   Mitch                   PM
0187636550       1002        980        M   801102               H
14           Miura                   Yuu                     PM
0197664120       1003        2003       M                        H
15           Davenport               Jason                   webmaster
0119556884       1003        1000       M   901212               H
16           Chen                    Wei-Wei                 manager
0195562100       1001        2300       F   780509               H
17           Fubuki                  Takahiro                PM
0165293886       2001        1400       M   781026  07-MAY-2010  H
18           Huxley                  John                    planner
01755231044      4001        1900       M           30-OCT-2007  H
19           Marquez                 Alvar                   sales rep
0185698550       4002        1800       M           18-NOV-2010  H
20           Blake                   William                 sales rep
01154112366      4002                   M           18-NOV-2006  H
20 rows selected.
```

departments Table

```
iSQL> select * from departments;
DNO          DNAME                         DEP_LOCATION       MGR_NO
-------------------------------------------------------------------------
1001         RESEARCH DEVELOPMENT DEPT 1   New York           16
1002         RESEARCH DEVELOPMENT DEPT 2   Sydney             13
1003         SOLUTION DEVELOPMENT DEPT     Osaka              14
2001         QUALITY ASSURANCE DEPT        Seoul              17
3001         CUSTOMERS SUPPORT DEPT        London             4
3002         PRESALES DEPT                 Peking             5
4001         MARKETING DEPT                Brasilia           8
4002         BUSINESS DEPT                 Palo Alto          7
8 rows selected.
```

customers Table

```
iSQL> select * from customers;
CNO               C_LASTNAME            C_FIRSTNAME
---------------------------------------------------------------------
CUS_JOB           CUS_TEL          SEX  BIRTH     POSTAL_CD
---------------------------------------------------------------------
ADDRESS
-------------------------------------------------------------
1                 Sanchez               Estevan
engineer          0514685282        M   720828   90021
2100 Exposition Boulevard Los Angeles USA
2                 Martin                Pierre
doctor            023242121         M   821215   V6T 1F2
4712 West 10th Avenue Vancouver BC Canada
3                 Morris                Gabriel
designer          023442542         M   811111   75010
D914 Puteaux Ile-de-France France
4                 Park                  Soo-jung
engineer          022326393         F   840305   609-735
Geumjeong-Gu Busan South Korea
5                 Stone                 James
webmaster         0233452141        M   821012   6060
142 Francis Street Western Australia AUS
6                 Dureault              Phil
WEBPD             025743215         M   810209   H1R-2W1
1000 Rue Rachel Est Montreal Canada
7                 Lalani                Yasmin
planner           023143366         F   821225   156772
176 Robinson Road Singapore
8                 Kanazawa              Tsubasa
PD                024721114         M   730801   141-0031
2-4-6 Nishi-Gotanda Shinagawa-ku Tokyo JP
9                 Yuan                  Ai
designer          0512543734        F   690211   200020
10th Floor No. 334 Jiujiang Road Shanghai
```

```
10                  Nguyen              Anh Dung
                    0516232256      M  790815  70000
8A Ton Duc Thang Street District 1 HCMC Vietnam
11                  Sato                Naoki
manager             027664545       M  810101  455-8205
3-23 Oye-cho Minato-ku Nagoya Aichi Japan
12                  Rodriguez           Aida
banker              023343214       F  810905  76152
3484 Taylor Street Dallas TX USA
13                  White               Crystal
engineer            022320119       F  801230  WC2B 4BM
12th Floor Five Kemble Street London UK
14                  Kim                 Cheol-soo
banker              024720112       M  660508  135-740
222-55 Samsung-dong Gangnam-gu Seoul Korea
15                  Fedorov             Fyodor
manager             0518064398      M  750625  50696
No 6 Leboh Ampang 50100 Kuala Lumpur Malaysia
16                  Lefebvre            Daniel
planner             027544147       M  761225  21004
Chaussee de Wavre 114a 1050 Brussels Belgium
17                  Yoshida             Daichi
                    023543541       M  811001  530-0100
2-7 3-Chome-Kita Tenjinbashi Kita-ku Osaka
18                  Zhang               Bao
engineer            024560207       F  840419  100008
2 Chaoyang Men Wai Street Chaoyang Beijing
19                  Pahlavi             Saeed
                    022371234       M  741231  20037
3300 L Street NW Washington DC USA
20                  Dubois              Alisee
webmaster           024560002       F  860405  1357
Chemin de Messidor 7-6 CH-1006 Lausanne Suisse
20 rows selected.
```

orders Table

```
iSQL> select * from orders;
ONO                 ORDER_DATE  ENO         CNO
------------------------------------------------------------------------
GNO        QTY         ARRIVAL_DATE PROCESSING
------------------------------------------------------
11290007               29-NOV-2011  12           3
A111100002  70            02-DEC-2011  C
11290011               29-NOV-2011  12           17
E111100001  1000          05-DEC-2011  D
11290100               29-NOV-2011  19           11
E111100001  500           07-DEC-2011  D
12100277               10-DEC-2011  19           5
D111100008  2500          12-DEC-2011  C
12300001               01-DEC-2011  19           1
D111100004  1000          02-JAN-2012  P
```

```
12300002              29-DEC-2011  12           2
C111100001  300         02-JAN-2012  P
12300003              29-DEC-2011  20           14
E111100002  900         02-JAN-2012  P
12300004              30-DEC-2011  20           15
D111100002  1000        02-JAN-2012  P
12300005              30-DEC-2011  19           4
D111100008  4000        02-JAN-2012  P
12300006              30-DEC-2011  20           13
A111100002  20          02-JAN-2012  P
12300007              30-DEC-2011  12           7
D111100002  2500        02-JAN-2012  P
12300008              30-DEC-2011  20           11
D111100011  300         02-JAN-2012  P
12300009              30-DEC-2011  20           19
D111100003  500         02-JAN-2012  P
12300010              30-DEC-2011  19           16
D111100010  2000        02-JAN-2012  P
12300011              30-DEC-2011  20           15
C111100001  1000        02-JAN-2012  P
12300012              30-DEC-2011  12           3
E111100012  1300        02-JAN-2012  P
12300013              30-DEC-2011  20           6
C111100001  5000        02-JAN-2012  P
12300014              30-DEC-2011  12           12
F111100001  800         02-JAN-2012  P
12310001              31-DEC-2011  20           15
A111100002  50          09-DEC-2011  O
12310002              31-DEC-2011  12           10
D111100008  10000       03-JAN-2012  O
12310003              31-DEC-2011  20           18
E111100009  1500        03-JAN-2012  O
12310004              31-DEC-2011  19           5
E111100010  5000        08-JAN-2012  O
12310005              31-DEC-2011  20           14
E111100007  940         03-JAN-2012  O
12310006              31-DEC-2011  20           2
D111100004  500         03-JAN-2012  O
12310007              31-DEC-2011  12           19
E111100012  1400        03-JAN-2012  O
12310008              31-DEC-2011  19           1
D111100003  100         03-JAN-2012  O
12310009              31-DEC-2011  12           5
E111100013  500         03-JAN-2012  O
12310010              31-DEC-2011  20           6
D111100010  1500        03-JAN-2012  O
12310011              31-DEC-2011  19           15
E111100012  10000       03-JAN-2012  O
12310012              31-DEC-2011  19           1
C111100001  250         03-JAN-2012  O
30 rows selected.
```

goods Table

```
iSQL> SELECT * FROM goods;
GOODS.GNO       GOODS.GNAME       GOODS.GOODS_LOCATION       GOODS.STOCK
---------------------------------------------------------------------------
GOODS.PRICE
-------------
A111100001      IM-300            AC0001                     1000
78000
A111100002      IM-310            DD0001                     100
98000
B111100001      NT-H5000          AC0002                     780
35800
C111100001      IT-U950           FA0001                     35000
7820.55
C111100002      IT-U200           AC0003                     1000
9455.21
D111100001      TM-H5000          AC0004                     7800
12000
D111100002      TM-T88            BF0001                     10000
72000


D111100003      TM-L60            BF0002                     650
45100
D111100004      TM-U950           DD0002                     8000
96200
D111100005      TM-U925           AC0005                     9800
23000
D111100006      TM-U375           EB0001                     1200
57400
D111100007      TM-U325           EB0002                     20000
84500
D111100008      TM-U200           AC0006                     61000
10000
D111100009      TM-U300           DD0003                     9000
50000
D111100010      TM-U590           DD0004                     7900
36800
D111100011      TM-U295           FA0002                     1000
45600
E111100001      M-T245            AC0007                     900
2290.54
E111100002      M-150             FD0001                     4300
7527.35
E111100003      M-180             BF0003                     1000
2300.55
E111100004      M-190G            CE0001                     88000
5638.76
E111100005      M-U310            CE0002                     11200
1450.5
E111100006      M-T153            FD0002                     900
```

```
2338.62
E111100007        M-T102            BF0004                    7890
966.99
E111100008        M-T500            EB0003                    5000
1000.54
E111100009        M-T300            FA0003                    7000
3099.88
E111100010        M-T260            AC0008                    4000
9200.5
E111100011        M-780             AC0009                    9800
9832.98
E111100012        M-U420            CE0003                    43200
3566.78
E111100013        M-U290            FD0003                    12000
1295.44
F111100001        AU-100            AC0010                    10000
100000
30 rows selected.
```

dual Table

```
iSQL> SELECT * FROM dual;
DUAL.X
------------
X
1 row selected.
```