

优炫数据库系统目录手册 2.1



UXSINO
优炫软件

优炫数据库系统目录手册 2.1

版权 © 2016-2023 北京优炫软件股份有限公司

法律声明

优炫数据库管理系统(简称: UXDB) 是由北京优炫软件股份有限公司开发并发布的一款商业性数据库管理系统。

优炫数据库管理系统 (UXDB) 的一切知识产权以及与该软件产品相关的所有信息内容, 包括但不限于: 文字表述及其组合、图标、图饰、图表、色彩、界面设计、版面框架、有关数据、及电子文档等均属北京优炫软件股份有限公司所有。本软件及其文档的任何使用、复制、修改、出租、传播、销售及分发等行为均须经北京优炫软件股份有限公司书面许可。

凡侵犯北京优炫软件股份有限公司知识产权的行为, 北京优炫软件股份有限公司将依法追究其法律责任。

本声明的最终解释权归属于北京优炫软件股份有限公司。



和其他优炫公司商标均为北京优炫软件股份有限公司的商标。

本文档提及的其他所有商标或注册商标, 由各自的所有人拥有。

注意

由于产品版本安装或其他原因, 本文档内容会不定期进行更新。除非另有约定, 本文档仅作为使用指导, 本文档中的所有陈述、信息和建议不构成任何明示或暗示的担保。

北京优炫软件股份有限公司 (总部)

- 地址: 北京市海淀区学院南路62号中关村资本大厦11层 (邮编: 100081)
 - 网址: <http://www.uxsino.com>
 - 邮箱: <uxdb_support@uxsino.com>
 - 电话: 010-82886998
 - 传真: 010-82886338
 - 服务热线: 400-650-7837
-

目录

前言	xi
1. 文档目的	xi
2. 文档对象	xi
3. 修改记录	xi
1. 系统目录	1
1.1. 概述	1
1.2. ux_aggregate	3
1.3. ux_am	5
1.4. ux_amop	5
1.5. ux_amproc	6
1.6. ux_attrdef	6
1.7. ux_attribute	7
1.8. ux_authid	10
1.9. ux_auth_members	11
1.10. ux_cast	11
1.11. ux_class	12
1.12. ux_collation	16
1.13. ux_constraint	16
1.14. ux_conversion	19
1.15. ux_database	19
1.16. ux_db_role_setting	20
1.17. ux_default_acl	21
1.18. ux_depend	21
1.19. ux_description	23
1.20. ux_enum	24
1.21. ux_event_trigger	24
1.22. ux_extension	25
1.23. ux_foreign_data_wrapper	25
1.24. ux_foreign_server	26
1.25. ux_foreign_table	26
1.26. ux_index	27
1.27. ux_inherits	29
1.28. ux_init_privs	29
1.29. ux_language	30
1.30. ux_largeobject	31
1.31. ux_largeobject_metadata	31
1.32. ux_namespace	32
1.33. ux_opclass	32
1.34. ux_operator	32
1.35. ux_opfamily	33
1.36. ux_partitioned_table	34
1.37. ux_pltemplate	35
1.38. ux_policy	35
1.39. ux_proc	36
1.40. ux_publication	39
1.41. ux_publication_rel	40
1.42. ux_range	40
1.43. ux_replication_origin	40
1.44. ux_rewrite	41
1.45. ux_seclabel	42
1.46. ux_sequence	42

1.47.	ux_shdepend	42
1.48.	ux_shdescription	43
1.49.	ux_shseclabel	44
1.50.	ux_statistic	44
1.51.	ux_statistic_ext	46
1.52.	ux_statistic_ext_data	47
1.53.	ux_subscription	47
1.54.	ux_subscription_rel	48
1.55.	ux_tablespace	48
1.56.	ux_transform	49
1.57.	ux_trigger	49
1.58.	ux_ts_config	51
1.59.	ux_ts_config_map	51
1.60.	ux_ts_dict	51
1.61.	ux_ts_parser	52
1.62.	ux_ts_template	52
1.63.	ux_type	53
1.64.	ux_user_mapping	58
1.65.	系统视图	59
1.66.	ux_available_extensions	60
1.67.	ux_available_extension_versions	60
1.68.	ux_config	61
1.69.	ux_cursors	61
1.70.	ux_file_settings	62
1.71.	ux_group	62
1.72.	ux_hba_file_rules	63
1.73.	ux_indexes	63
1.74.	ux_locks	64
1.75.	ux_matviews	66
1.76.	ux_policies	67
1.77.	ux_prepared_statements	67
1.78.	ux_prepared_xacts	68
1.79.	ux_publication_tables	69
1.80.	ux_replication_origin_status	69
1.81.	ux_replication_slots	69
1.82.	ux_roles	71
1.83.	ux_rules	71
1.84.	ux_seclabels	72
1.85.	ux_sequences	72
1.86.	ux_settings	73
1.87.	ux_shadow	75
1.88.	ux_stats	76
1.89.	ux_stats_ext	77
1.90.	ux_tables	79
1.91.	ux_timezone_abbrevs	79
1.92.	ux_timezone_names	79
1.93.	ux_user	80
1.94.	ux_user_mappings	80
1.95.	ux_views	81
1.96.	兼容oracle视图	81
1.97.	all_arguments	84
1.98.	all_col_comments	85
1.99.	all_col_privs	86
1.100.	all_cons_columns	86

1.101.	all_constraints	86
1.102.	all_ind_columns	88
1.103.	all_indexes	88
1.104.	all_objects	91
1.105.	all_sequences	92
1.106.	all_source	93
1.107.	all_synonyms	93
1.108.	all_tab_cols	93
1.109.	all_tab_columns	95
1.110.	all_tab_comments	97
1.111.	all_tab_privs	98
1.112.	all_tables	98
1.113.	all_trigger_cols	101
1.114.	all_triggers	101
1.115.	all_users	102
1.116.	all_views	102
1.117.	db_files	103
1.118.	dba_arguments	103
1.119.	dba_col_comments	105
1.120.	dba_col_privs	105
1.121.	dba_cons_columns	105
1.122.	dba_constraints	106
1.123.	dba_ind_columns	107
1.124.	dba_indexes	108
1.125.	dba_objects	111
1.126.	dba_role_privs	111
1.127.	dba_roles	112
1.128.	dba_sequences	112
1.129.	dba_source	112
1.130.	dba_synonyms	113
1.131.	dba_tab_cols	113
1.132.	dba_tab_columns	115
1.133.	dba_tab_comments	117
1.134.	dba_tab_privs	118
1.135.	dba_tables	118
1.136.	dba_tablespace	121
1.137.	dba tablespaces	121
1.138.	dba_trigger_cols	122
1.139.	dba_triggers	122
1.140.	dba_users	123
1.141.	dba_views	124
1.142.	user_arguments	124
1.143.	user_col_comments	126
1.144.	user_col_privs	126
1.145.	user_cons_columns	127
1.146.	user_constraints	127
1.147.	user_ind_columns	129
1.148.	user_indexes	129
1.149.	user_objects	132
1.150.	user_role_privs	133
1.151.	user_sequences	133
1.152.	user_source	134
1.153.	user_synonyms	134
1.154.	user_tab_cols	134

1.155.	user_tab_columns	136
1.156.	user_tab_comments	138
1.157.	user_tab_privs	139
1.158.	user_tables	139
1.159.	user_tablespace	142
1.160.	user_tablespaces	142
1.161.	user_trigger_cols	143
1.162.	user_triggers	143
1.163.	user_users	144
1.164.	user_views	145
1.165.	V\$DATABASE	145
1.166.	V\$INSTANCE	146
1.167.	V\$LOCK	147
1.168.	V\$LOCKED_OBJECT	148
1.169.	V\$PARAMETER	149
1.170.	V\$SESSION	150
1.171.	V\$SYSSTAT	156
2.	数据目录	157
3.	日志文件	159
3.1.	ux_log	159
3.2.	ux_xlog	160
3.2.1.	设置	160
3.2.2.	检查点	164
3.2.3.	归档	165
3.3.	ux_clog	165

表格清单

1. 文档更新记录	xi
1.1. 系统目录	1
1.2. ux_aggregate的列	3
1.3. ux_am的列	5
1.4. ux_amop的列	5
1.5. ux_amproc的列	6
1.6. ux_attrdef的列	7
1.7. ux_attribute的列	7
1.8. ux_authid的列	10
1.9. ux_auth_members的列	11
1.10. ux_cast的列	11
1.11. ux_class的列	12
1.12. ux_collation的列	16
1.13. ux_constraint的列	17
1.14. ux_conversion的列	19
1.15. ux_database的列	19
1.16. ux_db_role_setting的列	21
1.17. ux_default_acl的列	21
1.18. ux_depend的列	22
1.19. ux_description的列	23
1.20. ux_enum的列	24
1.21. ux_event_trigger的列	24
1.22. ux_extension的列	25
1.23. ux_foreign_data_wrapper的列	25
1.24. ux_foreign_server的列	26
1.25. ux_foreign_table的列	27
1.26. ux_index的列	27
1.27. ux_inherits的列	29
1.28. ux_init_privs列	30
1.29. ux_language的列	30
1.30. ux_largeobject的列	31
1.31. ux_largeobject_metadata的列	31
1.32. ux_namespace的列	32
1.33. ux_opclass的列	32
1.34. ux_operator的列	33
1.35. ux_opfamily的列	33
1.36. ux_partitioned_table列	34
1.37. ux_pltemplate的列	35
1.38. ux_policy列	35
1.39. ux_proc的列	36
1.40. ux_publication的列	39
1.41. ux_publication_rel列	40
1.42. ux_range的列	40
1.43. ux_replication_origin的列	41
1.44. ux_rewrite的列	41
1.45. ux_seclabel的列	42
1.46. ux_sequence的列	42
1.47. ux_shdepend的列	43
1.48. ux_shdescription的列	44
1.49. ux_shseclabel的列	44
1.50. ux_statistic的列	45

1.51.	ux_statistic_ext的列	46
1.52.	ux_statistic_ext_data Columns	47
1.53.	ux_subscription的列	47
1.54.	ux_subscription_rel的列	48
1.55.	ux_tablespace的列	48
1.56.	ux_transform的列	49
1.57.	ux_trigger的列	49
1.58.	ux_ts_config的列	51
1.59.	ux_ts_config_map的列	51
1.60.	ux_ts_dict的列	52
1.61.	ux_ts_parser的列	52
1.62.	ux_ts_template的列	52
1.63.	ux_type的列	53
1.64.	<i>typcategory</i> 编码	58
1.65.	ux_user_mapping的列	58
1.66.	系统视图	59
1.67.	ux_available_extensions的列	60
1.68.	ux_available_extension_versions的列	60
1.69.	ux_config列	61
1.70.	ux_cursors的列	61
1.71.	ux_file_settings的列	62
1.72.	ux_group的列	63
1.73.	ux_hba_file_rules的列	63
1.74.	ux_indexes的列	64
1.75.	ux_locks的列	64
1.76.	ux_matviews的列	67
1.77.	ux_policies的列	67
1.78.	ux_prepared_statements的列	68
1.79.	ux_prepared_xacts的列	68
1.80.	ux_publication_tables的列	69
1.81.	ux_replication_origin_status的列	69
1.82.	ux_replication_slots的列	69
1.83.	ux_roles的列	71
1.84.	ux_rules的列	72
1.85.	ux_seclabels的列	72
1.86.	ux_sequences的列	72
1.87.	ux_settings的列	73
1.88.	ux_shadow的列	75
1.89.	ux_stats的列	76
1.90.	ux_stats_ext的列	78
1.91.	ux_tables的列	79
1.92.	ux_timezone_abbrevs的列	79
1.93.	ux_timezone_names的列	80
1.94.	ux_user的列	80
1.95.	ux_user_mappings的列	80
1.96.	ux_views的列	81
1.97.	兼容oracle视图	81
1.98.	all_arguments的列	84
1.99.	all_col_comments的列	85
1.100.	all_col_privs的列	86
1.101.	all_cons_columns的列	86
1.102.	all_constraints的列	86
1.103.	all_ind_columns的列	88

1.104.	all_indexes的列	89
1.105.	all_objects的列	92
1.106.	all_sequences的列	92
1.107.	all_source的列	93
1.108.	all_synonyms的列	93
1.109.	all_tab_cols的列	93
1.110.	all_tab_columns的列	96
1.111.	all_tab_comments的列	98
1.112.	all_tab_privs的列	98
1.113.	all_tables 的列	98
1.114.	all_trigger_cols的列	101
1.115.	all_triggers的列	101
1.116.	all_users的列	102
1.117.	all_views的列	102
1.118.	db_files的列	103
1.119.	dba_arguments的列	103
1.120.	dba_col_comments的列	105
1.121.	dba_col_privs的列	105
1.122.	dba_cons_columns的列	105
1.123.	dba_constraints的列	106
1.124.	dba_ind_columns的列	107
1.125.	dba_indexes的列	108
1.126.	dba_objects的列	111
1.127.	dba_role_privs的列	112
1.128.	dba_roles的列	112
1.129.	dba_sequences的列	112
1.130.	dba_source的列	113
1.131.	dba_synonyms的列	113
1.132.	dba_tab_cols 的列	113
1.133.	dba_tab_columns 的列	115
1.134.	dba_tab_comments的列	117
1.135.	dba_tab_privs的列	118
1.136.	dba_tables 的列	118
1.137.	dba_tablespace的列	121
1.138.	dba_tablespaces的列	121
1.139.	dba_trigger_cols的列	122
1.140.	dba_triggers的列	122
1.141.	dba_users的列	123
1.142.	dba_views的列	124
1.143.	user_arguments的列	124
1.144.	user_col_comments的列	126
1.145.	user_col_privs的列	126
1.146.	user_cons_columns的列	127
1.147.	user_constraints的列	127
1.148.	user_ind_columns的列	129
1.149.	user_indexes的列	129
1.150.	user_objects的列	132
1.151.	user_role_privs的列	133
1.152.	user_sequences的列	133
1.153.	user_source的列	134
1.154.	user_synonyms的列	134
1.155.	user_tab_cols的列	134
1.156.	user_tab_columns的列	137
1.157.	user_tab_comments的列	139

1.158.	user_tab_privs的列	139
1.159.	user_tables的列	139
1.160.	user_tablespace的列	142
1.161.	user_tablespace的列	143
1.162.	user_trigger_cols的列	143
1.163.	user_triggers的列	144
1.164.	user_users的列	144
1.165.	user_views的列	145
1.166.	V\$DATABASE的列	146
1.167.	V\$INSTANCE的列	146
1.168.	V\$LOCK的列	147
1.169.	V\$LOCKED_OBJECT的列	148
1.170.	V\$PARAMETER的列	149
1.171.	V\$SESSION的列	150
1.172.	V\$SYSSTAT的列	156
2.1.	UXDATA的内容	157
3.1.	ux_log常用日志配置	159

前言

1. 文档目的

本文档介绍了优炫数据库的系统目录、数据目录和日志文件，为相关技术人员和用户提供了必要的参考。

2. 文档对象

- 技术支持工程师
- 维护工程师
- 优炫数据库用户

3. 修改记录

修改记录累积了每次文档更新的说明。最新版本的文档包含以前所有文档版本的更新内容。

表 1. 文档更新记录

工具版本	发布日期	修改说明
2.1.1.5C	2023-01-06	第一次正式发布。

第 1 章 系统目录

系统目录是关系型数据库存放模式元数据的地方，比如表和列的信息，以及内部统计信息等。UXsinoDB的系统目录就是普通表。你可以删除并重建这些表、增加列、插入和更新数值，然后彻底把你的系统搞垮。通常情况下，我们不应该手工修改系统目录，通常有SQL命令可以做这些事情。（例如，**CREATE DATABASE**向 `ux_database`表插入一行一并且实际上在磁盘上创建该数据库。）。有几种特别深奥的操作例外，但是随着时间的流逝其中的很多也可以用 SQL 命令来完成，因此对系统目录直接修改的需求也越来越小。

1.1. 概述

表 [1.1 “系统目录”](#)列出了系统目录。每个目录更详细的文档见后文。

大多数系统目录都是在数据库创建的过程中从模版数据库中拷贝过来的，因此都是数据库相关的。少数目录在物理上是在一个集簇的所有数据库间中共享的，这些将在每一个目录单独的描述中介绍。

表 1.1. 系统目录

目录名	用途
ux_aggregate	聚集函数
ux_am	关系访问方法
ux_amop	访问方法操作符
ux_amproc	访问方法支持函数
ux_attrdef	列默认值
ux_attribute	表列（“属性”）
ux_authid	认证标识符（角色）
ux_auth_members	认证标识符成员关系
ux_cast	转换（数据类型转换）
ux_class	表、索引、序列、视图（“关系”）
ux_collation	排序规则（locale信息）
ux_constraint	检查约束、唯一约束、主键约束、外键约束
ux_conversion	编码转换信息
ux_database	本数据库集簇中的数据库
ux_db_role_setting	每角色和每数据库的设置
ux_default_acl	对象类型的默认权限
ux_depend	数据库对象间的依赖
ux_description	数据库对象上的描述或注释
ux_enum	枚举标签和值定义
ux_event_trigger	事件触发器
ux_extension	已安装扩展
ux_foreign_data_wrapper	外部数据包装器定义
ux_foreign_server	外部服务器定义

目录名	用途
ux_foreign_table	外部表信息
ux_index	索引信息
ux_inherits	表继承层次
ux_init_privs	对象初始特权
ux_language	编写函数的语言
ux_largeobject	大对象的数据页
ux_largeobject_metadata	大对象的元数据
ux_namespace	模式
ux_opclass	访问方法操作符类
ux_operator	操作符
ux_opfamily	访问方法操作符族
ux_partitioned_table	表的分区键的信息
ux_pltemplate	过程语言的模板数据
ux_policy	行安全策略
ux_proc	函数和过程
ux_publication	用于逻辑复制的发布
ux_publication_rel	发布映射的关系
ux_range	范围类型的信息
ux_replication_origin	已注册的复制源
ux_rewrite	查询重写规则
ux_seclabel	数据库对象上的安全标签
ux_sequence	有关序列的信息
ux_shdepend	共享对象上的依赖
ux_shdescription	共享对象上的注释
ux_shseclabel	共享数据库对象上的安全标签
ux_statistic	规划器统计
ux_statistic_ext	扩展的规划器统计信息（定义）
ux_statistic_ext_data	扩展的规划器统计信息（已构建的统计信息）
ux_subscription	逻辑复制订阅
ux_subscription_rel	订阅的关系状态
ux_tablespace	本数据库集簇内的表空间
ux_transform	转换（将数据类型转换为过程语言需要的形式）
ux_trigger	触发器
ux_ts_config	文本搜索配置
ux_ts_config_map	文本搜索配置的记号映射
ux_ts_dict	文本搜索字典
ux_ts_parser	文本搜索分析器

目录名	用途
ux_ts_template	文本搜索模板
ux_type	数据类型
ux_user_mapping	将用户映射到外部服务器

1.2. ux_aggregate

目录ux_aggregate存储关于聚集函数的信息。聚集函数是对一个数值集合（典型的是每个匹配查询条件的行中的同一个列的值）进行操作的函数，它返回从这些值中计算出的一个数值。典型的聚集函数是 `sum`、`count`和`max`。ux_aggregate里的每个项都是一个ux_proc项的扩展。ux_proc项记载该聚集的名字、输入和输出数据类型，以及其他一些和普通函数类似的信息。

表 1.2. ux_aggregate的列

名称	类型	引用	描述
<i>aggfnoid</i>	regproc	ux_proc.oid	聚集函数在ux_proc中的OID
<i>aggkind</i>	char		聚集类型： <code>n</code> 表示“普通”聚集， <code>o</code> 表示“有序集”聚集，或者 <code>h</code> 表示“假想集”聚集
<i>aggnumdirectargs</i>	int2		一个有序集或者假想集聚集的直接（非聚集）参数的数量，一个可变数组算作一个参数。如果等于 <code>pronargs</code> ，该聚集必定是可变的并且该可变数组描述聚集参数以及最终直接参数。对于普通聚集总是为零。
<i>aggtransfn</i>	regproc	ux_proc.oid	转移函数
<i>aggfinalfn</i>	regproc	ux_proc.oid	最终函数（如果没有就为零）
<i>aggcombinefn</i>	regproc	ux_proc.oid	结合函数（如果没有就为零）
<i>aggserialfn</i>	regproc	ux_proc.oid	序列化函数（如果没有就为零）
<i>aggdeserialfn</i>	regproc	ux_proc.oid	反序列化函数（如果没有就为零）
<i>aggmtransfn</i>	regproc	ux_proc.oid	用于移动聚集模式的向前转移函数（如果没有就为零）
<i>aggminvtransfn</i>	regproc	ux_proc.oid	用于移动聚集模式的反向转移函数（如果没有就为零）

名称	类型	引用	描述
<i>aggmfinalfn</i>	regproc	ux_proc.oid	用于移动聚集模式的最终函数（如果没有就为零）
<i>aggfinalextra</i>	bool		为真则向 <i>aggfinalfn</i> 传递额外的哑参数
<i>aggmfinalextra</i>	bool		为真则向 <i>aggmfinalfn</i> 传递额外的哑参数
<i>aggfinalmodify</i>	char		<i>aggfinalfn</i> 是否修改传递状态值：如果是只读则为r，如果不能在 <i>aggfinalfn</i> 之后应用 <i>aggtransfn</i> 则为s，如果它修改该值则为w
<i>aggmfinalmodify</i>	char		和 <i>aggfinalmodify</i> 类似，但是用于 <i>aggmfinalfn</i>
<i>aggstortop</i>	oid	ux_operator.oid	相关联的排序操作符（如果没有则为0）
<i>aggtranstype</i>	oid	ux_type.oid	聚集函数的内部转移（状态）数据的数据类型
<i>aggtransspace</i>	int4		转移状态数据的近似平均尺寸（字节），或者为零表示使用一个默认估算值
<i>aggmtranstype</i>	oid	ux_type.oid	聚集函数用于移动聚集欧氏的内部转移（状态）数据的数据类型（如果没有则为零）
<i>aggmtransspace</i>	int4		转移状态数据的近似平均尺寸（字节），或者为零表示使用一个默认估算值
<i>agginitval</i>	text		转移状态的初始值。这是一个文本域，它包含初始值的外部字符串表现形式。如果这个域为空，则转移状态值从空值开始。
<i>aggminitval</i>	text		用于移动聚集模式的转移状态初值。这是一个文本域，它包含了以其文本字符串形式表达的初值。如果这个域为空，则转移状态值从空值开始。

新的聚集函数可通过CREATE AGGREGATE命令注册。

1.3. ux_am

目录ux_am存储关于关系访问方法的信息。系统支持的每种访问方法在这个目录中都有一行。目前只有表和索引拥有访问方法。

表 1.3. ux_am的列

名称	类型	引用	描述
<i>oid</i>	oid		行标识符
<i>amname</i>	name		访问方法的名字
<i>amhandler</i>	regproc	ux_proc.oid	负责提供有关该访问方法信息的处理器函数的OID
<i>amtype</i>	char		t = 表(包括物化视图), i = 索引.

注意

ux_am包含很多额外的列以表示索引访问方法的性质。那些数据现在只有在C代码级别才是直接可见的。不过，系统中增加了ux_index_column_has_property()和一些相关函数来允许SQL查询检查索引访问方法的性质。

1.4. ux_amop

目录ux_amop存储关于与访问方法操作符族相关的操作符信息。对于一个操作符族中的每一个成员即操作符都在这个目录中有一行。一个成员可以是一个搜索操作符或者一个排序操作符。一个操作符可以出现在多个族中，但在同一个组中既不能出现在多个搜索位置也不能出现在多个排序位置（虽然不太可能出现，但是允许一个操作符同时用于搜索和排序目的）。

表 1.4. ux_amop的列

名称	类型	引用	描述
<i>oid</i>	oid		行标识符
<i>amopfamily</i>	oid	ux_opfamily.oid	这个项所在的操作符族
<i>amoplefttype</i>	oid	ux_type.oid	操作符的左手输入数据类型
<i>amoprightrighttype</i>	oid	ux_type.oid	操作符的右手输入数据类型
<i>amopstrategy</i>	int2		操作符策略号
<i>amoppurpose</i>	char		操作符目的，s表示搜索，o表示排序
<i>amopopr</i>	oid	ux_operator.oid	操作符的OID
<i>amopmethod</i>	oid	ux_am.oid	使用此操作符族的索引访问方法

名称	类型	引用	描述
<i>amopsortfamily</i>	oid	ux_opfamily.oid	如果是一个排序操作符，该项会根据这个 B 树操作符族排序，如果是一个搜索操作符则为 0

一个“搜索”操作符项意味着该操作符族的一个索引可以被搜索来查找所有满足如下条件的行：**WHERE *indexed_column operator constant***。显然，这样的操作符必须返回boolean，且它的左手输入类型必须匹配索引列的数据类型。

一个“排序”操作符项意味着该操作符族的一个索引可以被扫描来返回以如下顺序排列的行：**ORDER BY *indexed_column operator constant***。这样一个操作符能够返回任何可排序数据类型，尽管它的左手输入类型必须匹配索引列的数据类型。**ORDER BY**的准确语义由*amopsortfamily*列指定，它必须引用一个适合于操作符结果类型的B树操作符族。

注意

目前，一个排序操作符的排序顺序被假设为其引用的操作符族的默认值，即ASC NULLS LAST。未来可能会通过增加额外的列来显式地指定排序选项。

一个项的*amopmethod*必须和它所包含的操作符族的opfmethod相匹配（这里包括*amopmethod*是一个为了性能原因而故意对目录结构做的反规范化）。此外，*amoplefttype*和*amoprighttype*也必须匹配被引用的ux_operator项的*oprleft*和*oprright*域。

1.5. ux_amproc

目录ux_amproc存储关于访问方法操作符族相关的支持函数。属于一个操作符族的每一个支持函数在这个目录中都有一行。

表 1.5. ux_amproc的列

名称	类型	引用	描述
<i>oid</i>	oid		行标识符
<i>amprocfamily</i>	oid	ux_opfamily.oid	使用这个项的操作符族
<i>amproclefttype</i>	oid	ux_type.oid	相关操作符的左手输入数据类型
<i>amprocrighttype</i>	oid	ux_type.oid	相关操作符的右手输入数据类型
<i>amprocnum</i>	int2		支持过程编号
<i>amproc</i>	regproc	ux_proc.oid	过程的OID

*amproclefttype*和*amprocrighttype*列的通常解释是它们标识了一个特定支持过程所支持的操作符的左右输入类型。对于某些访问方法它们和支持过程本身的输入数据类型相匹配，而对其他的则不会匹配。对于一个索引有一个“默认”支持过程的概念，这些支持过程的*amproclefttype*和*amprocrighttype*都等于索引操作符类的*opcintype*。

1.6. ux_attrdef

`ux_attrdef`存储列的默认值。列的主要信息存储在`ux_attribute`。只有那些显式指定了一个默认值的列才会在这个目录中有一个项。

表 1.6. `ux_attrdef`的列

名称	类型	引用	描述
<i>oid</i>	oid		行标识符
<i>adrelid</i>	oid	ux_class.oid	该列所属的表
<i>adnum</i>	int2	ux_attribute.attnum	列号
<i>adbin</i>	ux_node_tree		列的默认值，以 <code>nodeToString()</code> 表示。 用 <code>ux_get_expr(adbin, adrelid)</code> 将其转换为SQL表达式。

1.7. `ux_attribute`

目录`ux_attribute`存储有关表列的信息。数据库中的每一个表的每一个列都恰好在`ux_attribute`中有一行。（这其中也会有索引的属性项，并且事实上所有具有`ux_class`项的对象在这里都有属性项） entries.）

术语属性等同于列，这里使用它只是出于历史原因。

表 1.7. `ux_attribute`的列

名称	类型	引用	描述
<i>attrelid</i>	oid	ux_class.oid	列所属的表
<i>attname</i>	name		列名
<i>atttypid</i>	oid	ux_type.oid	列的数据类型
<i>attstattarget</i>	int4		<i>attstattarget</i> 控制由ANALYZE对此列收集的统计信息的细节层次。0值表示不会收集任何统计信息。一个负值则说明直接使用系统默认的目标。正值的确切含义取决于数据类型。对于标量数据类型， <i>attstattarget</i> 既是要收集的“最常见值”的目标号，也是要创建的柱状图容器的目标号。
<i>attlen</i>	int2		本列类型的 <code>ux_type.typelen</code> 一个拷贝
<i>attnum</i>	int2		列的编号。一般列从1开始向上编号。系统列

名称	类型	引用	描述
			(如 <code>ctid</code>)则拥有(任意)负值编号。
<code>attndims</code>	int4		如果该列是一个数组类型,这里就是其维度数;否则为0。(在目前一个数组的维度数并不被强制,因此任何非零值都能有效地表明“这是一个数组”。)
<code>attcacheoff</code>	int4		在存储中总是为-1,但是当被载入到一个内存中的行描述符后,这里可能会被更新为属性在行内的偏移
<code>atttypmod</code>	int4		<code>atttypmod</code> 记录了在表创建时提供的类型相关数据(例如一个varchar列的最大长度)。它会被传递给类型相关的输入函数和长度强制函数。对于那些不需要 <code>atttypmod</code> 的类型,这个值通常总是为-1。
<code>attbyval</code>	bool		该列类型的 <code>ux_type.typbyval</code> 的一个拷贝
<code>attstorage</code>	char		通常是该列类型的 <code>ux_type.typstorage</code> 的一个拷贝。对于可TOAST的数据类型,这可以在列创建后被修改以控制存储策略。
<code>attalign</code>	char		该列类型的 <code>ux_type.typalign</code> 的一个拷贝
<code>attnotnull</code>	bool		这表示一个非空约束。
<code>atthasdef</code>	bool		该列有一个默认表达式或生成的表达式,在此情况下在 <code>ux_attrdef</code> 目录中会有一个对应项来真正定义该表达式。 (检查 <code>attgenerated</code> 以确定是默认还是生成的表达式。)
<code>attmissing</code>	bool		该列在行中完全缺失时会用到这个列的值,如果在行创建之后增加一

名称	类型	引用	描述
			个有非易失DEFAULT值的列，就会发生这种情况。实际使用的值被存放在 <code>attmissingval</code> 列中。
<code>attidentity</code>	char		如果是一个零字节(")，则不是一个标识列。否则， <code>a</code> = 总是生成， <code>d</code> = 默认生成。
<code>attgenerated</code>	char		如果为零字节(")，则不是生成的列。否则， <code>s</code> = stored。（将来可能会添加其他值。）
<code>attisdropped</code>	bool		该列被删除且不再有效。一个删除的列仍然物理存在于表中，但是会被分析器忽略并因此无法通过SQL访问。
<code>attislocal</code>	bool		该列是由关系本地定义的。注意一个列可以同时是本地定义和继承的。
<code>attinhcount</code>	int4		该列的直接祖先的编号。一个具有非零编号祖先的列不能被删除或者重命名。
<code>attcollation</code>	oid	ux_collation.oid	该列被定义的排序规则，如果该列不是一个可排序数据类型则为0。
<code>attacl</code>	aclitem[]		列级访问权限
<code>attoptions</code>	text[]		属性级选项，以“keyword=value”形式的字符串
<code>attfdwoptions</code>	text[]		属性级的外部数据包装器选项，以“keyword=value”形式的字符串
<code>attmissingval</code>	anyarray		这个列中是一个含有一个元素的数组，其中的值被用于该列在行中完全缺失时，如果在行创建之后增加一个有非易失DEFAULT值的列，就会发生这种情况。只有当 <code>atthasmissing</code> 为真

名称	类型	引用	描述
			时才使用这个值。如果没有值则该列为空。

在一个被删除的列的`ux_attribute`的项中，`attypid`被重置为0，但`attlen`以及其他从`ux_type`拷贝的域仍然有效。这种安排用于处理一种情况，即被删除列的数据类型后来被删除，并且因此不再有相应的`ux_type`行。`attlen`和其他域可以被用来解释表的一行的内容。

1.8. `ux_authid`

目录`ux_authid`包含关于数据库授权标识符（角色）的信息。角色把“用户”和“组”的概念包含在内。一个用户实际上就是一个`rolcanlogin`标志被设置的角色。任何角色（不管`rolcanlogin`设置与否）都能够把其他角色作为成员，参见[ux_auth_members](#)。

由于这个目录包含口令，它不能是公共可读的。[ux_roles](#)是在`ux_authid`上的一个公共可读视图，它隐去了口令域。

由于用户标识符是集簇范围的，`ux_authid`在一个集簇的所有数据库之间共享：在一个集簇中只有一份`ux_authid`拷贝，而不是每个数据库一份。

表 1.8. `ux_authid`的列

名字	类型	描述
<i>oid</i>	oid	行标识符
<i>rolname</i>	name	角色名
<i>rolsuper</i>	bool	角色是否拥有超级用户权限
<i>rolinherit</i>	bool	如果本角色是另一个角色的成员，本角色是否自动另一个角色的权限
<i>rolcreatorole</i>	bool	角色是否能创建更多角色
<i>rolcreatedb</i>	bool	角色是否能创建数据库
<i>rolcanlogin</i>	bool	角色是否能登录。即该角色是否能够作为初始会话授权标识符
<i>rolreplication</i>	bool	角色是一个复制角色。复制角色可以启动复制连接并且创建和删除复制槽。
<i>rolbypassrls</i>	bool	角色是否可以绕过所有的行级安全性策略。
<i>rolconlimit</i>	int4	对于可以登录的角色，本列设置该角色可以同时发起最大连接数。-1表示无限制。
<i>rolpassword</i>	text	口令（可能被加密过），如果没有口令则为空。格式取决于使用的加密方法的形式。
<i>rolvaliduntil</i>	timestampz	口令过期时间（只用于口令鉴定），如果永不过期则为空

对于一个MD5加密的口令，*rolpassword*列将由字符串md5后面跟上一个32字符的十六进制MD5哈希值构成。MD5哈希值将是该用户的口令串接上它们的用户名。例如，如果用户joe的口令是xyzyz，则UXsinoDB将存储xyzyzjoe的md5哈希。

如果口令采用SCRAM-SHA-256加密，它的格式是：

SCRAM-SHA-256\$<iteration count>:<salt>\$<StoredKey>:<ServerKey>

其中*salt*、*StoredKey*和*ServerKey*是Base64编码格式。这种格式与RFC 5803说明的格式相同。

不遵守上述格式的口令被假定为未加密。

1.9. ux_auth_members

目录ux_auth_members展示了角色之间的成员关系。允许任何无环的关系集合。

由于用户标识符是集簇范围的，ux_auth_members在一个集簇的所有数据库之间共享：在一个集簇中只有一份ux_auth_members拷贝，而不是每个数据库一份。

表 1.9. ux_auth_members的列

名称	类型	引用	描述
<i>roleid</i>	oid	ux_authid.oid	拥有成员的角色的ID
<i>member</i>	oid	ux_authid.oid	<i>roleid</i> 的成员角色的ID
<i>grantor</i>	oid	ux_authid.oid	授权此成员关系的角色的ID
<i>admin_option</i>	bool		如果 <i>member</i> 能把 <i>roleid</i> 的成员关系授予他人，则为真

1.10. ux_cast

目录ux_cast存储数据类型转换路径，包括内建的和用户定义的类型。

需要注意的是，ux_cast并不表示系统知道如何执行的所有类型转换，它只包括哪些不能从某些普通规则推导出的转换。例如，一个域及其基类型之间的转换并未显式地在ux_cast中展示。另一个重要的例外是“自动 I/O转换造型”，它们通过数据类型自己的I/O函数来转换成（或者转换自）text或其他字符串类型，这些转换也没有显式地在ux_cast中表示。

表 1.10. ux_cast的列

名称	类型	引用	描述
<i>oid</i>	oid		行标识符
<i>castsource</i>	oid	ux_type.oid	源数据类型的OID
<i>casttarget</i>	oid	ux_type.oid	目标数据类型的OID
<i>castfunc</i>	oid	ux_proc.oid	执行该转换的函数的OID。如果该转换方法不需要一个函数则存储0。

名称	类型	引用	描述
<i>castcontext</i>	char		指示该转换能被调用的环境。 e 表示仅能作为一个显式转换（使用CAST或::语法）。 a 表示在赋值给目标列时隐式调用，和显式调用一样。 i 表示在表达式中隐式调用，和其他转换一样。
<i>castmethod</i>	char		指示转换如何被执行。 f 表明使用 <i>castfunc</i> 中指定的函数。 i 表明使用输入/输出函数。 b 表明该类型是二进制可转换的，因此不需要转换。

在ux_cast里列出的类型转换函数必须总是以转换的源类型作为它的第一个参数类型，并且返回转换的目标类型作为它的结果类型。一个类型转换函数最多有三个参数。如果出现了第二个参数，必须是integer类型；它接受与目标类型关联的修饰词，如果没有，就是 -1。如果出现了第三个参数，那么必须是boolean类型；如果该类型转换是一种明确的转换，那么它接受true，否则接受false。

在ux_cast里创建一条源类型和目标类型相同的记录是合理的，只要相关联的函数接受多过一个参数。这样的记录代表“长度转换函数”，它们把该类型的值转换为对特定的类型合法的值。

如果一个ux_cast的项有着不同的原类型和目标类型，并且有一个接收多于一个参数的函数，那么它会在一个步骤中完成从一种类型到另外一种类型的转换并应用一个长度转换。如果没有这样的项，使用一个类型修改器的转换涉及两个步骤，一个是在数据类型之间转换，另外一个应用修改器。

1.11. ux_class

目录ux_class记录表和几乎所有具有列或者像表的东西。这包括索引（但还要参见ux_index）、序列（但还要参见ux_sequence）、视图、物化视图、组合类型和TOAST表，参见relkind。下面，当我们提及所有这些类型的对象时我们使用“关系”。并非所有列对于所有关系类型都有意义。

表 1.11. ux_class的列

名称	类型	引用	描述
<i>oid</i>	oid		行标识符
<i>relname</i>	name		表、索引、视图等的名字
<i>relnamespace</i>	oid	ux_namespace.oid	包含该关系的名字空间的OID
<i>reltype</i>	oid	ux_type.oid	可能存在的表行类型所对应数据类型的OID（对索引为0，索引没有ux_type项）

名称	类型	引用	描述
<i>reloftype</i>	oid	ux_type.oid	对于有类型的表，为底层组合类型的OID，对于其他所有关系为0
<i>relowner</i>	oid	ux_authid.oid	关系的拥有者
<i>relam</i>	oid	ux_am.oid	如果这是一个表或者索引，表示索引使用的访问方法（堆、B树、哈希等）
<i>relfilenode</i>	oid		该关系的磁盘文件的名称，0表示这是一个“映射”关系，其磁盘文件名取决于低层状态
<i>reltablespace</i>	oid	ux_tablespace.oid	该关系所存储的表空间。如果为0，使用数据库的默认表空间。 （如果关系无磁盘文件时无意义）
<i>relpages</i>	int4		该表磁盘表示的尺寸，以页面计（页面尺寸为BLCKSZ）。这只是一个由规划器使用的估计值。它 被VACUUM、ANALYZE以及一些DDL命令 （如CREATE INDEX）所更新。
<i>reltuples</i>	float4		表中的存活行数。这只是一个由规划器使用的估计值。它 被VACUUM、ANALYZE以及一些DDL命令 （如CREATE INDEX）所更新。
<i>relallvisible</i>	int4		在表的可见性映射表中被标记为全可见的页数。这只是一个由规划器使用的估计值。它 被VACUUM、ANALYZE以及一些DDL命令 （如CREATE INDEX）所更新。
<i>reltoastrelid</i>	oid	ux_class.oid	与该表相关联的TOAST表的OID，如果没有则为0。TOAST表将大属性“线外”存储在一个二级表中。

名称	类型	引用	描述
<i>relhasindex</i>	bool		如果这是一个表并且其上建有（或最近建有）索引则为真
<i>relisshared</i>	bool		如果该表在集簇中的所有数据库间共享则为真。只有某些系统目录（如ux_database）是共享的。
<i>relpersistence</i>	char		p = 永久表, u = 无日志表, t = 临时表
<i>relkind</i>	char		r = 普通表, i = 索引, S = 序列, t = TOAST表, v = 视图, m = 物化视图, c = 组合类型, f = 外部表, p = 分区表, I = 分区索引
<i>relnatts</i>	int2		关系中用户列的数目（系统列不计算在内）。在ux_attribute中必须有这么多对应的项。另请参阅ux_attribute.attnum。
<i>relchecks</i>	int2		表上CHECK约束的数目, 参见 ux_constraint 目录
<i>relhasrules</i>	bool		如果表有（或曾有）规则则为真, 参见 ux_rewrite 目录
<i>relhastriggers</i>	bool		如果表有（或曾有）触发器则为真, 参见 ux_trigger 目录
<i>relhassubclass</i>	bool		如果表或者索引有（或曾有）任何继承子女则为真
<i>relrowsecurity</i>	bool		如果表上启用了行级安全性则为真, 参见 ux_policy 目录
<i>relforcerowsecurity</i>	bool		如果行级安全性（启用时）也适用于表拥有者则为真, 参见 ux_policy 目录
<i>relispopulated</i>	bool		如果表已被填充则为真（对于所有关系该列都为真, 但对于某些物化视图却不是）

名称	类型	引用	描述
<i>relreplident</i>	char		用来为行形成“replica identity”的列： d = 默认（主键，如果存在）， n = 无， f = 所有列 i = 索引的 <i>indisreplident</i> 被设置或者为默认
<i>relispartition</i>	bool		如果表或索引是一个分区，则为真
<i>relrewrite</i>	oid	ux_class.oid	对于在要求表重写的DDL操作期间被写入的新关系，这个域包含原始关系的OID，否则为0。那种状态仅在内部可见，对于一个用户可见的关系这个域应该从不包含不是0的值。
<i>relfrozenxid</i>	xid		在此之前的所有事务ID在表中已经被替换为一个永久的（“冻结的”）事务ID。这用于跟踪表是否需要被清理，以便阻止事务ID回卷或者允许ux_xact被收缩。如果该关系不是一个表则为0（InvalidTransactionId）。
<i>relminmxid</i>	xid		在此之前的多事务ID在表中已经被替换为一个事务ID。这被用于跟踪表是否需要被清理，以阻止多事务ID回卷或者允许ux_multixact被收缩。如果关系不是一个表则为0（InvalidMultiXactId）。
<i>relacl</i>	aclitem[]		访问权限
<i>reloptions</i>	text[]		访问方法相关的选项，以“keyword=value”字符串形式
<i>relpartbound</i>	ux_node_tree		如果表示一个分区（见 <i>relispartition</i> ），分区边界的内部表达

ux_class中的一些逻辑标志被以一种懒惰的方式维护：在正确状态时它们被保证为真，但是当条件不再为真时它们并不会被立刻重置为假。例如，*relhasindex*由CREATE INDEX设置，但它从不会被DROP INDEX清除。作为替代，VACUUM会在找到无索引表后清除其*relhasindex*。这种安排避免了竞争条件并且提高了并发性。

1.12. ux_collation

目录`ux_collation`描述了可用的排序规则，其本质是从一个SQL名字到操作系统`locale`分类的映射。

表 1.12. `ux_collation`的列

名称	类型	引用	描述
<i>oid</i>	oid		行标识符
<i>collname</i>	name		排序规则名字（在每一个名字空间和编码中唯一）
<i>collnamespace</i>	oid	ux_namespace.oid	包含该排序规则的名字空间的OID
<i>collowner</i>	oid	ux_authid.oid	排序规则的拥有者
<i>collprovider</i>	char		排序规则的提供者：d = 数据库默认，c = libc，i = icu
<i>collisdeterministic</i>	bool		排序规则是确定性的吗？
<i>collencoding</i>	int4		该排序规则可应用的编码，-1表示它可用于任何编码
<i>collcollate</i>	name		该排序规则对象的LC_COLLATE
<i>collctype</i>	name		该排序规则对象的LC_CTYPE
<i>collversion</i>	text		排序规则的提供者相关的版本。这是在排序规则创建时记录下来的，并且在使用排序规则时会被检查以检测可能导致数据损坏的排序规则定义的改变。

注意在这个目录中的唯一键是（`collname`、`collencoding`、`collnamespace`），不仅仅是（`collname`，`collnamespace`）。所有`collencoding`不等于当前数据库编码或-1的编码规则通常都会被UXsinoDB忽略，且禁止创建和`collencoding` = -1的项重名的项。因此使用一个受限的SQL名字（`schema.name`）来标识一个排序规则是足够的，即使这根据目录定义是不唯一的。以这种方式定义这个目录的原因是`initdb`会在集簇初始化时使用系统上所有可用的`locale`填充这个目录，所以它必须能够为所有可能在集簇中使用的编码保持项。

在`template0`数据库中，创建与数据库编码不匹配的编码是有用的，因为它们可以匹配后面从`template0`克隆的数据库的编码。这在目前必须手动完成。

1.13. ux_constraint

目录ux_constraint存储表上的检查、主键、唯一、外键和排他约束（列约束也不会被特殊对待。每一个列约束都等同于某种表约束。）。非空约束不在这里，而是在ux_attribute目录中表示。

用户定义的约束触发器（使用CREATE CONSTRAINT TRIGGER创建）也会在这个表中产生一项。

域上的检查约束也存储在这里。

表 1.13. ux_constraint的列

名称	类型	引用	描述
<i>oid</i>	oid		行标识符
<i>conname</i>	name		约束名字（不需要唯一！）
<i>connamespace</i>	oid	ux_namespace.oid	包含此约束的名字空间的OID
<i>contype</i>	char		c = 检查约束， f = 外键约束， p = 主键约束， u = 唯一约束， t = 约束触发器， x = 排他约束
<i>condeferrable</i>	bool		该约束是否能被延迟？
<i>condeferred</i>	bool		该约束是否默认被延迟？
<i>convalidated</i>	bool		此约束是否被验证过？当前对于外键和检查约束只能是假
<i>conrelid</i>	oid	ux_class.oid	该约束所在的表，如果不是表约束则为0
<i>contypid</i>	oid	ux_type.oid	该约束所在的域，如果不是域约束则为0
<i>conindid</i>	oid	ux_class.oid	如果该约束是唯一、主键、外键或排他约束，此列表示支持此约束的索引，否则为0
<i>conparentid</i>	oid	ux_constraint.oid	如果这是一个分区中的约束，则是父分区表中对应的约束；否则为0
<i>confrelid</i>	oid	ux_class.oid	如果此约束是一个外键约束，此列为被引用的表，否则为0
<i>confupdtype</i>	char		外键更新动作代码： a = 无动作， r = 限制， c = 级联， n = 置空， d = 置为默认值
<i>confdeltype</i>	char		外键删除动作代码： a = 无动作， r = 限制，

名称	类型	引用	描述
			c = 级联, n = 置空, d = 置为默认值
<i>confmatchtype</i>	char		外键匹配类型: f = 完全, p = 部分, s = 简单
<i>conislocal</i>	bool		此约束是定义在关系本地。注意一个约束可以同时是本地定义和继承。
<i>coninhcount</i>	int4		此约束的直接继承祖先数目。一个此列非零的约束不能被删除或重命名。
<i>connoinherit</i>	bool		为真表示此约束被定义在关系本地。它是一个不可继承约束。
<i>conkey</i>	int2[]	ux_attribute.attnum	如果是一个表约束（包括外键但不包括约束触发器），此列是被约束列的列表
<i>confkey</i>	int2[]	ux_attribute.attnum	如果是一个外键，此列是被引用列的列表
<i>confpeqop</i>	oid[]	ux_operator.oid	如果是一个外键，此列是用于PK = FK比较的等值操作符的列表
<i>conppeqop</i>	oid[]	ux_operator.oid	如果是一个外键，此列是用于PK = PK比较的等值操作符的列表
<i>conffeqop</i>	oid[]	ux_operator.oid	如果是一个外键，此列是用于FK = FK比较的等值操作符的列表
<i>conexcllop</i>	oid[]	ux_operator.oid	如果是一个排他约束，此列是没列排他操作符的列表
<i>conbin</i>	ux_node_tree		如果是一个检查约束，此列是表达式的一个内部表示。建议使用 <code>ux_get_constraintdef()</code> 提取检查约束的定义。

在一个排他约束的情况下，*conkey*只对约束元素是单一列引用时有用。对于其他情况，*conkey*为0且必须查阅相关索引来发现被约束的表达式（*conkey*因此和`ux_index.indkey`具有相同的内容）。

注意

`ux_class.relchecks`需要和每个关系在此目录中的检查约束数量保持一致。

1.14. ux_conversion

目录ux_conversion描述编码转换函数。

表 1.14. ux_conversion的列

名称	类型	引用	描述
<i>oid</i>	oid		行标识符
<i>conname</i>	name		转换的名字（在一个名字空间内唯一）
<i>connamespace</i>	oid	ux_namespace.oid	包含此转换的名字空间的OID
<i>conowner</i>	oid	ux_authid.oid	转换的拥有者
<i>conforencoding</i>	int4		源编码ID
<i>contoencoding</i>	int4		目标编码ID
<i>conproc</i>	regproc	ux_proc.oid	转换函数
<i>condefault</i>	bool		如果这是默认转换则为真

1.15. ux_database

目录ux_database存储有关可用数据库的信息。数据库通过CREATE DATABASE命令创建。

和大部分系统目录不同，ux_database是在集簇的所有数据库之间共享的：在一个集簇中只有一份ux_database拷贝，而不是每个数据库一份。

表 1.15. ux_database的列

名称	类型	引用	描述
<i>oid</i>	oid		行标识符
<i>datname</i>	name		数据库名字
<i>datdba</i>	oid	ux_authid.oid	数据库的拥有者，通常是创建它的用户
<i>encoding</i>	int4		此数据库的字符编码的编号 (ux_encoding_to_char() 可 将此编号转换成编码的 名字)
<i>datcollate</i>	name		此数据库的LC_COLLATE
<i>datctype</i>	name		此数据库的LC_CTYPE
<i>datistemplate</i>	bool		如果为真，则此数据库 可被任何具 有CREATEDB特权的 用户克隆；如果为假， 则只有 超级用户或者 该数据库的属主能够克 隆它。

名称	类型	引用	描述
<i>datallowconn</i>	bool		如果为假则没有人能连接到这个数据库。这可以用来保护template0数据库不被修改。
<i>datconnlimit</i>	int4		设置能够连接到这个数据库的最大并发连接数。-1表示没有限制。
<i>datlastsysoid</i>	oid		数据库中最后一个系统OID，对ux_dump特别有用
<i>datfrozensid</i>	xid		在此之前的所有事务ID在数据库中已经被替换为一个永久的（“冻结的”）事务ID。这用于跟踪数据库是否需要被清理，以便组织事务ID回环或者允许ux_xact被收缩。它是此数据库中所有表的ux_class.relfrozensid值的最小值。
<i>datminmxid</i>	xid		在此之前的所有多事务ID在数据库中已经被替换为一个事务ID。这用于跟踪数据库是否需要被清理，以便组织事务ID回环或者允许ux_multixact被收缩。它是此数据库中所有表的ux_class.relminmxid值的最小值。
<i>dattablespace</i>	oid	ux_tablespace.oid	此数据库的默认表空间。在此数据库中，所有ux_class.reltablespace为0的表都将被存储在这个表空间中，尤其是非共享系统目录都会在其中。
<i>datacl</i>	aclitem[]		访问权限

1.16. ux_db_role_setting

目录 ux_db_role_setting为每一个角色和数据库组合记录被设置到运行时配置变量的默认值。

和大部分系统目录不同，ux_db_role_setting是在集簇的所有数据库之间共享的：在一个集簇中只有一份ux_db_role_setting拷贝，而不是每个数据库一份。

表 1.16. `ux_db_role_setting`的列

名称	类型	引用	描述
<i>setdatabase</i>	oid	ux_database.oid	此设置可用的数据库OID，如果不与具体数据库相关则为0
<i>setrole</i>	oid	ux_authid.oid	此设置可用的角色OID，如果不与具体角色相关则为0
<i>setconfig</i>	text[]		运行时配置变量的默认值

1.17. `ux_default_acl`

目录`ux_default_acl`存储要被分配给新创建对象的初始权限。

表 1.17. `ux_default_acl`的列

名称	类型	引用	描述
<i>oid</i>	oid		行标识符
<i>defaclrole</i>	oid	ux_authid.oid	与此项相关的角色的OID
<i>defaclnamespace</i>	oid	ux_namespace.oid	与此项相关的名字空间的OID，如果没有则为0
<i>defaclobjtype</i>	char		此项适合的对象类型： r = 关系（表、视图）， S = 序列， f = 函数， T = 类型， n = 方案
<i>defaclacl</i>	aclitem[]		此类对象在创建时应用有的访问权限

一个`ux_default_acl`项展示了对要分配给属于一个指定用户的对象的初始权限。当前有两类项：`defaclnamespace = 0`的“全局”项和引用一个特殊模式的“每方案”项。如果一个全局项存在，则它重载该对象类型的普通hard-wired默认权限。一个每模式项如果存在，表示权限将被加入到全局或hard-wired默认权限中。

注意当在另一个表中的一个ACL项为空时，它用来表示其对象的hard-wired默认权限，而不是当时可能在`ux_default_acl`中的任何权限。只有在对象创建期间才会查阅`ux_default_acl`。

1.18. `ux_depend`

目录`ux_depend`记录数据库对象之间的依赖关系。这些信息允许**DROP**命令查找必须被**DROP CASCADE**删除的其他对象，或者在**DROP RESTRICT**情况下阻止删除。

另请参阅[ux_shdepend](#)，它对在一个数据库集群中共享的对象之间的依赖提供了相似的功能。

表 1.18. ux_depend的列

名称	类型	引用	描述
<i>classid</i>	oid	ux_class.oid	依赖对象所在的系统目录OID
<i>objid</i>	oid	任意OID列	指定依赖对象的OID
<i>objsubid</i>	int4		对于一个表列，这里是列号（ <i>objid</i> 和 <i>classid</i> 指表本身）。对于所有其他对象类型，此列为0。
<i>refclassid</i>	oid	ux_class.oid	被引用对象所在的系统目录的OID
<i>refobjid</i>	oid	任意OID列	指定被引用对象的OID
<i>refobjsubid</i>	int4		对于一个表列，这里是列号（ <i>refobjid</i> 和 <i>refclassid</i> 指表本身）。对于所有其他对象类型，此列为0。
<i>deptype</i>	char		定义此依赖关系语义的一个代码，见文本

在所有情况下，一个ux_depend项表明被引用对象不能在没有删除其依赖对象的情况下被删除。但是，其中也有多种依赖类型，由*deptype*标识：

DEPENDENCY_NORMAL (n)

在独立创建的对象之间的一个普通关系。依赖对象可以在不影响被依赖对象的情况下被删除。被引用对象只能通过指定CASCADE被删除，在这种情况下依赖对象也会被删除。例子：一个表列对于其数据类型有一个普通依赖。

DEPENDENCY_AUTO (a)

依赖对象可以被独立于被依赖对象删除，且应该在被引用对象被删除时自动被删除（不管在RESTRICT或CASCADE模式）。例子：一个表上的一个命名约束应该被设置为自动依赖于表，这样在表被删除后它也会消失。

DEPENDENCY_INTERNAL (i)

依赖对象作为被引用对象创建过程的一部分被创建，并且只是其内部实现的一部分。不允许直接DROP所依赖的对象（而是告诉用户对引用对象发出DROP操作）。无论是否指定了CASCADE，DROP被引用的对象都将导致自动删除从属对象。如果由于删除了对某些其他对象的依赖关系而不得不删除依赖对象，则其删除将转换为对所引用对象的删除，因此依赖对象的NORMAL和AUTO依赖关系的行为就像它们是所引用对象的依赖关系。示例：视图的ON SELECT规则使其在内部依赖于视图，以防止在视图保留时将其删除。规则的依赖关系（例如它引用的表）就好像他们是视图的依赖关系。

DEPENDENCY_PARTITION_PRI (P)

DEPENDENCY_PARTITION_SEC (S)

依赖对象被作为被引用对象创建过程的一部分创建，并且确实是其内部实现的一部分。但是，不像INTERNAL，有多个这样的引用对象。除非删除了这些引用对象中的至少一个对

象，否则不得删除依赖对象；如果其中任何一个被删除，则不管是否指定了CASCADE，都应删除依赖对象。也不像INTERNAL，依赖对象所依赖的某些其他对象的删除不会导致任何分区引用的对象的自动删除。因此，如果删除没有通过其他路径级联到这些对象中的至少一个，它会被拒绝。（大多数情况下，依赖对象与至少一个分区引用对象共享所有非分区的依赖关系，因此此限制不会导致阻止任何级联的删除。）主分区和辅助分区的依赖关系表现相同，除了主分区依赖关系倾向于错误消息；因此，分区相关的对象应该有一个主分区依赖关系和一个或多个辅助分区依赖关系。注意到分区依赖关系是任何对象所正常拥有的依赖关系的补充，而不是替代。这简化了ATTACH/DETACH PARTITION操作：只要添加或删除分区的依赖关系。例如：子分区索引与其所基于的分区表和父分区索引是分区相关的，因此只要其中一个删除，则子分区索引就消失，否则，就不消失。父索引上的依赖关系是主要的，故如果用户试图删除子分区索引，错误消息反而会建议删除父索引（不是表）。

DEPENDENCY_EXTENSION (e)

依赖对象是作为扩展的被引用对象的一个成员（参见[ux_extension](#)）。依赖对象可以通过被引用对象上的DROP EXTENSION来删除。在功能上，这种依赖类型和一个INTERNAL依赖的作用相同，其存在只是为了清晰和简化ux_dump。

DEPENDENCY_AUTO_EXTENSION (x)

依赖对象不是作为被引用对象的扩展的成员（因此不应该被ux_dump忽略），但是没有该扩展它又无法工作，因此如果删除了扩展，则该依赖对象应自动删除。该依赖对象也可以独立删除。功能上，该依赖关系类型与AUTO依赖相同，但是为了清晰起见和简化ux_dump，将其分开。

DEPENDENCY_PIN (p)

没有依赖对象，这种类型的项是一个信号，用于说明系统本身依赖于被引用对象，并且该对象永远不能被删除。这种类型的项只能被initdb创建。而此种项的依赖对象的列都为0。

在未来可能会需要其他依赖类型。

要注意的是，两个对象很有可能由不止一个ux_depend条目来链接。例如子分区索引有一个依赖于其关联的分区表的分区类型的依赖关系和依赖于该表索引的每一列的自动依赖关系。此类情形表示多重依赖关系语义的并集，依赖对象的删除可以没有CASCADE，如果其任一依赖关系满足自动删除的条件。相反地，关于哪些对象必须一起删除的所有依赖关系的限制必须满足。

1.19. ux_description

目录ux_description存储对每一个数据库对象可选的描述（注释）。描述可以通过COMMENT操作，并可使用uxsql的\d命令查看。在ux_description的初始内容中提供了很多内建系统对象的描述。

参见[ux_shdescription](#)，它对在一个数据库集簇中共享的对象的描述提供了相似的功能。

表 1.19. ux_description的列

名称	类型	引用	描述
<i>objoid</i>	oid	任意OID列	描述所属对象的OID
<i>classoid</i>	oid	ux_class.oid	对象所述的系统目录的OID
<i>objsubid</i>	int4		对于一个表列上的一个注释，这里是列号

名称	类型	引用	描述
			(<i>objoid</i> 和 <i>classoid</i> 指表本身)。对所有其他对象类型，此列为0。
<i>description</i>	text		作为该对象描述的任意文本

1.20. ux_enum

`ux_enum`目录包含每一个枚举类型的项，其中包括了值和标签。一个给定枚举值的内部表示实际上是它在`ux_enum`中的相关行的OID。

表 1.20. `ux_enum`的列

名称	类型	引用	描述
<i>oid</i>	oid		行标识符
<i>enumtypid</i>	oid	ux_type.oid	包含此枚举值的 <code>ux_type</code> 项的OID
<i>enumsortorder</i>	float4		此枚举值在其枚举类型 中的排序位置
<i>enumlabel</i>	name		此枚举值的文本标签

`ux_enum`行的OID值遵循一种特殊的规则：即OID的数值被保证按照其枚举类型的排序顺序进行排序。即如果两个偶数OID属于同一枚举类型，较小的OID必然具有较小的*enumsortorder*值。奇数OID值不需要遵循排序顺序。这种规则使得枚举比较例程在很多常见情况下可以避免系统目录查找。创建和修改枚举类型的例程将尝试尽可能地为枚举值分配偶数OID。

当一个枚举类型被创建后，其成员会被分配排序位置1..*n*。但后面增加的成员可能会被分配负值或者分数值的*enumsortorder*。对于这些值的唯一要求是它们必须被正确地排序且和保持唯一。

1.21. ux_event_trigger

目录`ux_event_trigger`存储事件触发器。

表 1.21. `ux_event_trigger`的列

名称	类型	引用	描述
<i>evtname</i>	name		触发器名（必须唯一）
<i>evtevent</i>	name		此触发器触发的事件的标识符
<i>evtowner</i>	oid	ux_authid.oid	事件触发器的拥有者
<i>evtfoid</i>	oid	ux_proc.oid	将被调用的函数
<i>evtenabled</i>	char		控制事件触发器触发的 <code>session_replication_role</code> 模式。 O = 触发器 在“origin”和“local”模 式触发， D = 触发器

名称	类型	引用	描述
			被禁用， R = 触发器在“replica”模式触发， A = 触发器总是触发。
<i>evttags</i>	text[]		此触发器将触发的命令标签。如果为空，此触发器的触发不受命令标签的限制。

1.22. ux_extension

目录ux_extension存储有关已安装扩展的信息。

表 1.22. ux_extension的列

名称	类型	引用	描述
<i>oid</i>	oid		行标识符
<i>extname</i>	name		扩展的名字
<i>extowner</i>	oid	ux_authid.oid	扩展的拥有者
<i>extnamespace</i>	oid	ux_namespace.oid	包含此扩展的导出对象的模式
<i>extrelocatable</i>	bool		如果扩展可被重定位到另一个模式则为真
<i>extversion</i>	text		扩展的版本名字
<i>extconfig</i>	oid[]	ux_class.oid	扩展的配置表的regclass项的OID数组，如果没有配置表则为NULL
<i>extcondition</i>	text[]		扩展的配置表的WHERE子句过滤条件的数组，如果没有则为NULL

注意和大部分具有一个“namespace”列的模式不同，*extnamespace*不是用来表示扩展属于该模式。扩展的名字从不用模式进行限定。*extnamespace*表明该模式包含了该扩展的大部分或全部对象。如果*extrelocatable*为真，则该模式事实上必须包含属于此扩展的全部模式限定的对象。

1.23. ux_foreign_data_wrapper

目录ux_foreign_data_wrapper存储外部数据包装器定义。外部数据包装器是一种访问位于外部服务器上数据的机制。

表 1.23. ux_foreign_data_wrapper的列

名称	类型	引用	描述
<i>oid</i>	oid		行标识符

名称	类型	引用	描述
<i>fdwname</i>	name		外部数据包装器的名字
<i>fdwowner</i>	oid	ux_authid.oid	外部数据包装器的所有者
<i>fdwhandler</i>	oid	ux_proc.oid	指一个负责为外部数据包装器提供执行例程的处理函数。如果没有提供处理函数则为0
<i>fdwvalidator</i>	oid	ux_proc.oid	指一个负责检查传给外部数据包装器的选项的有效性的验证函数，包括外部服务器选项以及使用外部数据包装器的用户映射。如果没有提供验证函数则为0
<i>fdwacl</i>	aclitem[]		访问权限
<i>fdwoptions</i>	text[]		外部数据包装器特定选项，以“keyword=value”字符串形式

1.24. ux_foreign_server

目录ux_foreign_server存储外部服务器定义。外部服务器定义了外部数据的来源，例如一个远程服务器。外部服务器通过外部数据包装器来访问。

表 1.24. ux_foreign_server的列

名称	类型	引用	描述
<i>oid</i>	oid		行标识符
<i>srvname</i>	name		外部服务器的名称
<i>srvowner</i>	oid	ux_authid.oid	外部服务器的所有者
<i>srvfdw</i>	oid	ux_foreign_data_wrapper.oid	外部服务器的外部数据包装器的OID
<i>srvtype</i>	text		服务器的类型（可选）
<i>srvversion</i>	text		服务器的版本（可选）
<i>srvacl</i>	aclitem[]		访问权限
<i>srvoptions</i>	text[]		外部服务器特定选项，以“keyword=value”字符串形式

1.25. ux_foreign_table

目录ux_foreign_table包含关于外部表的辅助信息。一个外部表和普通表一样，主要由一个ux_class项表示。它的ux_foreign_table项包含外部表所特有的信息。

表 1.25. `ux_foreign_table`的列

名称	类型	引用	描述
<i>firelid</i>	oid	ux_class.oid	外部表的 <code>ux_class</code> 项的OID
<i>ftserver</i>	oid	ux_foreign_server.oid	外部表所在的外部服务器的OID
<i>ftoptions</i>	text[]		外部表选项，以“keyword=value”字符串形式

1.26. `ux_index`

目录`ux_index`包含关于索引的部分信息。其他信息大部分在`ux_class`中。

表 1.26. `ux_index`的列

名称	类型	引用	描述
<i>indexrelid</i>	oid	ux_class.oid	此索引的 <code>ux_class</code> 项的OID
<i>indrelid</i>	oid	ux_class.oid	此索引的基表的 <code>ux_class</code> 项的OID
<i>indnatts</i>	int2		索引中的总列数（与 <code>ux_class.relnatts</code> 重复），这个数目包括键和被包括的属性
<i>indnkeyatts</i>	int2		索引中键列的编号，不计入任何的内含列，它们只是被存储但不参与索引的语义
<i>indisunique</i>	bool		表示是否为唯一索引
<i>indisprimary</i>	bool		表示索引是否表示表的主键（如果此列为真， <i>indisunique</i> 也总是为真）
<i>indisexclusion</i>	bool		表示索引是否支持一个排他约束
<i>indimmediate</i>	bool		表示唯一性检查是否在插入时立即被执行（如果 <i>indisunique</i> 为假，此列无关）
<i>indisclustered</i>	bool		如果为真，表示表最后以此索引进行了聚簇
<i>indisvalid</i>	bool		如果为真，此索引当前可以用于查询。为假表示此索引可能不完整：它肯定还在被INSERT/UPDATE操作

名称	类型	引用	描述
			所修改，但它不能安全地被用于查询。如果索引是唯一索引，唯一性属性也不能被保证。
<i>indcheckxmin</i>	bool		如果为真，直到此 <code>ux_index</code> 行的 <i>xmin</i> 低于查询的 <code>TransactionXmin</code> 视界之前，查询都不能使用此索引，因为表可能包含具有它们可见的不相容行的损坏HOT链
<i>indisready</i>	bool		如果为真，表示此索引当前可以用于插入。为假表示索引必须被 INSERT/UPDATE 操作忽略。
<i>indislive</i>	bool		如果为假，索引正处于被删除过程中，并且必须被所有处理忽略（包括HOT安全的决策）
<i>indisreplident</i>	bool		如果为真，这个索引被选择为使用 ALTER TABLE ... REPLICA IDENTITY USING INDEX ... 的“ <code>replica identity</code> ”
<i>indkey</i>	int2vector	ux_attribute .attnum	这是一个 <i>indnatts</i> 值的数组，它表示了此索引索引的表列。例如一个 1 3 值可能表示表的第一和第三列组成了索引项。键列出现在非键（内含）列前面。数组中的一个 0 表示对应的索引属性是一个在表列上的表达式，而不是一个简单的列引用。
<i>indcollation</i>	oidvector	ux_collation .oid	对于索引键（ <i>indnkeyatts</i> 值）中的每一列，这包含要用于该索引的排序规则的 OID，如果该列不是一种可排序数据类型则为零。
<i>indclass</i>	oidvector	ux_opclass .oid	对于索引键中的每一列（ <i>indnkeyatts</i> 值），这里包含了要使用的操作

名称	类型	引用	描述
			符类的OID。详见 ux_opclass 。
<i>indoption</i>	int2vector		这是一个 <code>indnkeyatts</code> 值的数组，用于存储每列的标志位。位的意义由索引的访问方法定义。
<i>indexprs</i>	ux_node_tree		非简单列引用索引属性的表达式树（以 <code>nodeToString()</code> 形式）。对于 <code>indkey</code> 中每一个为0的项，这个列表中都有一个元素。如果所有的索引属性都是简单引用，此列为空。
<i>indpred</i>	ux_node_tree		部分索引谓词的表达式树（以 <code>nodeToString()</code> 形式）。如果不是部分索引，此列为空。

1.27. ux_inherits

目录`ux_inherits`记录有关表继承层次的信息。数据库中每一个直接父子关系在这里都有一项（非直接继承可以通过顺着项构成的链来决定）。

表 1.27. `ux_inherits`的列

名称	类型	引用	描述
<i>inhrelid</i>	oid	ux_class.oid	孩子表的OID
<i>inhparent</i>	oid	ux_class.oid	父表的OID
<i>inhseqno</i>	int4		如果一个孩子表有多于一个直接父表（多继承），这个数字说明了继承列被排列的顺序。计数从1开始。

1.28. ux_init_privs

目录`ux_init_privs`记录系统中对象的初始特权。数据库中每一个具有非默认（非-NULL）初始特权集合的对象都有一个条目在其中。

对象可以在系统初始化（`initdb`）时获得其初始特权，也可以在**CREATE EXTENSION**期间创建该对象并且在扩展脚本中用**GRANT**来设置对象的初始特权。注意系统将自动处理扩展脚本执行期间对特权的记录，扩展的作者们只需要在他们的脚本中使用**GRANT**以及**REVOKE**语句以便特权被记录下来。`privtype`列表示初始特权是被`initdb`设置还是在一次**CREATE EXTENSION**命令期间被设置。

具有被`initdb`设置的初始特权的对象的条目中`privtype`是'i'，而具有被**CREATE EXTENSION**设置的初始特权的对象的条目中`privtype`为'e'。

表 1.28. ux_init_privs列

名称	类型	引用	描述
<i>objoid</i>	oid	任何 OID 列	指定对象的 OID
<i>classoid</i>	oid	ux_class.oid	对象所在的系统目录的 OID
<i>objsubid</i>	int4		对于一个表列，这里是列编号 (<i>objoid</i> 和 <i>classoid</i> 指向表本身)。对于所有其他对象类型，这列为零。
<i>privtype</i>	char		定义这个对象初始特权类型的代码，见文字说明
<i>initprivs</i>	aclitem[]		初始的访问权限

1.29. ux_language

目录ux_language注册了可用于编写函数或存储过程的语言。

表 1.29. ux_language的列

名称	类型	引用	描述
<i>oid</i>	oid		行标识符
<i>lanname</i>	name		语言的名字
<i>lanowner</i>	oid	ux_authid.oid	语言的拥有者
<i>lanispl</i>	bool		内部语言为假（如 SQL），用户定义语言为真。当前，ux_dump 仍然使用这个列来决定要转储哪些语言，但在未来这可能会被一种不同的机制所取代。
<i>lanpltrusted</i>	bool		为真表示这是一种可信的语言，即它被相信不会向普通SQL执行环境之外的任何东西授予权限。只有超级用户可以在非可信语言中创建函数。
<i>lanplcallfoid</i>	oid	ux_proc.oid	对于非内部语言，此列引用语言处理器，它是一个特殊函数负责执行所有用这种语言编写的函数
<i>laninline</i>	oid	ux_proc.oid	此列引用一个负责执行“内联”匿名代码块

名称	类型	引用	描述
			的函数（D0块）。如果不支持内联块则为0。
<i>lanvalidator</i>	oid	ux_proc.oid	此列引用一个负责在函数创建时对其进行语法和可用性检查的语言验证函数。如果没有提供验证器则为0。
<i>lanacl</i>	aclitem[]		访问权限

1.30. ux_largeobject

目录ux_largeobject保存构成“大对象”的数据。一个大对象在被创建时会被分配一个OID。每个大对象被分解成段或“页”，以便小到可以被方便地作为行存储在ux_largeobject中。每页中的数据量被定义为LOBLKSIZE（目前是BLCKSZ/4或是2 kB）。

在UXsinoDB 9.0之前，大对象没有相关的权限结构。作为结果，ux_largeobject是公共可读的并且可以用来获得系统中所有大对象的OID（和内容）。但现在不是这样了，可使用[ux_largeobject_metadata](#)来获得大对象OID的列表。

表 1.30. ux_largeobject的列

名称	类型	引用	描述
<i>loid</i>	oid	ux_largeobject_metadata.oid	包含此页的大对象的标识符
<i>pageno</i>	int4		此页在它所属大对象中的页号（从0开始计）
<i>data</i>	bytea		实际存储在大对象中的数据。它从不会超过LOBLKSIZE字节，也可能更少。

ux_largeobject的每一行保存一个大对象的一个页的数据，从对象内部的字节偏移量（pageno * LOBLKSIZE）开始。现在的实现允许稀疏存储：页面可能丢失，并且可能比LOBLKSIZE字节短（即便不是最后一页）。一个大对象中丢失的区域会被读出为0。

1.31. ux_largeobject_metadata

目录ux_largeobject_metadata保持着与大对象有关的元数据。真正的大对象数据被存储在[ux_largeobject](#)中。

表 1.31. ux_largeobject_metadata的列

名称	类型	引用	描述
<i>oid</i>	oid		行标识符
<i>lomowner</i>	oid	ux_authid.oid	大对象的拥有者
<i>lomacl</i>	aclitem[]		访问权限

1.32. ux_namespace

目录ux_namespace存储名字空间。名字空间是SQL模式之下的结构：每个名字空间拥有一个独立的表、类型等的集合，且其中没有名字冲突。

表 1.32. ux_namespace的列

名称	类型	引用	描述
<i>oid</i>	oid		行标识符
<i>nspname</i>	name		名字空间的名字
<i>nspowner</i>	oid	ux_authid.oid	名字空间的拥有者
<i>nspacl</i>	aclitem[]		访问权限

1.33. ux_opclass

目录ux_opclass定义索引访问方法的操作符类。每一个操作符类定义了一种特定数据类型和一种特定索引访问方法的索引列的语义。一个操作符类实际上指定了一个特定的操作符族可以用于一个特定可索引列数据类型。该族中可用于索引列的操作符能够接受该列的数据类型作为它们的左输入。

表 1.33. ux_opclass的列

名称	类型	引用	描述
<i>oid</i>	oid		行标识符
<i>opcmethod</i>	oid	ux_am.oid	操作符类所属的索引访问方法
<i>opcname</i>	name		操作符类的名称
<i>opcnamespace</i>	oid	ux_namespace.oid	操作符类所属的名字空间
<i>opcowner</i>	oid	ux_authid.oid	操作符类的拥有者
<i>opcfamily</i>	oid	ux_opfamily.oid	包含此操作符类的操作符族
<i>opcintype</i>	oid	ux_type.oid	操作符类索引的数据类型
<i>opcdefault</i>	bool		如果此操作符类为 <i>opcintype</i> 的默认值则为真
<i>opkeytype</i>	oid	ux_type.oid	存储在索引中的数据的类型，如果值为0表示与 <i>opcintype</i> 相同

一个操作符类的*opcmethod*必须匹配包含它的操作符族的opfmethod。而且，对于任何给定的opcmethod和opcintype组合，只有不超过一个ux_opclass行的opcdefault值为真。

1.34. ux_operator

目录ux_operator存储关于操作符的信息。

表 1.34. `ux_operator`的列

名称	类型	引用	描述
<i>oid</i>	oid		行标识符
<i>oprname</i>	name		操作符的名称
<i>oprnamespace</i>	oid	ux_namespace.oid	操作符所属的名字空间的OID
<i>oprowner</i>	oid	ux_authid.oid	操作符的拥有者
<i>oprkind</i>	char		b = 中缀 (“并”), l = 前缀 (“左”), r = 后缀 (“右”)
<i>oprcanmerge</i>	bool		该操作符是否支持归并连接
<i>oprcanhash</i>	bool		该操作符是否支持哈希连接
<i>oprleft</i>	oid	ux_type.oid	左操作数类型
<i>oprright</i>	oid	ux_type.oid	右操作数类型
<i>oprresult</i>	oid	ux_type.oid	结果类型
<i>oprcom</i>	oid	ux_operator.oid	该操作符的交换子 (如果存在)
<i>oprnegate</i>	oid	ux_operator.oid	该操作符的否定 (如果存在)
<i>oprcode</i>	regproc	ux_proc.oid	实现该操作符的函数
<i>oprrest</i>	regproc	ux_proc.oid	该操作符的限制选择度估算函数
<i>oprjoin</i>	regproc	ux_proc.oid	该操作符的连接选择度估算函数

未用的列包含零值。例如，一个前缀操作符的`oprleft`为0。

1.35. `ux_opfamily`

目录`ux_opfamily`定义了操作符族。每一个操作符族是操作符和相关支持例程的集合，支持例程用于实现一个特定索引访问方法的语义。此外，按照访问方法指定的某种方式，一个族内的操作符都是“兼容的”。操作符族概念允许在索引中使用跨数据类型操作符，并可以使用访问方法语义的知识推导出。

表 1.35. `ux_opfamily`的列

名称	类型	引用	描述
<i>oid</i>	oid		行标识符
<i>opfmethod</i>	oid	ux_am.oid	操作符族适用的索引访问方法
<i>opfname</i>	name		操作符族的名字
<i>opfnamespace</i>	oid	ux_namespace.oid	操作符族所属的名字空间

名称	类型	引用	描述
<i>opfowner</i>	oid	ux_authid.oid	操作符族的拥有者

定义操作符族的主要信息不在它的`ux_opfamily`行，而是在相关的[ux_amop](#)、[ux_amproc](#)和[ux_opclass](#)行中。

1.36. `ux_partitioned_table`

目录`ux_partitioned_table`存放有关表如何被分区的信息。

表 1.36. `ux_partitioned_table`列

名称	类型	引用	描述
<i>partrelid</i>	oid	ux_class.oid	这个分区表的 <code>ux_class</code> 项的OID
<i>partstrat</i>	char		分区策略； h = 哈希分区表， l = 列表分区表， r = 范围分区表
<i>partnatts</i>	int2		分区键中的列数
<i>partdefid</i>	oid	ux_class.oid	这个分区表的默认分区的 <code>ux_class</code> 项的OID，如果这个分区表没有默认分区则为零。
<i>partattrs</i>	int2vector	ux_attribute.attnum	这是一个长度为 <code>partnatts</code> 值的数组，它指示哪些表列是分区键的组成部分。例如，值1 3表示第一个和第三个表列组成了分区键。这个数组中的零表示对应的分区键列是一个表达式而不是简单的列引用。
<i>partclass</i>	oidvector	ux_opclass.oid	对于分区键中的每一个列，这个域包含要使用的操作符类的OID。详见 ux_opclass 。
<i>partcollation</i>	oidvector	ux_opclass.oid	对于分区键中的每一个列，这个域包含要用于分区的排序规则的OID，如果该列不是一种可排序数据类型则为零。
<i>partexprs</i>	ux_node_tree		非简单列引用的分区键列的表达式树（以 <code>nodeToString()</code> 的表达方式）。这是一个列表， <code>partattrs</code> 中每一个零项都有一个元素。

名称	类型	引用	描述
			如果所有分区键列都是简单列引用，则这个域为空。

1.37. ux_pltemplate

目录ux_pltemplate存储了过程语言的“模板”信息。一个语言的模板允许我们在一个特定数据库中以一个简单的**CREATE LANGUAGE**命令创建语言，而不需要指定实现细节。

和大部分系统目录不同，ux_pltemplate是在集簇的所有数据库之间共享的：在一个集簇中只有一份ux_pltemplate拷贝，而不是每个数据库一份。这使得在每个需要的数据库中都可以访问该信息。

表 1.37. ux_pltemplate的列

名称	类型	描述
<i>tplname</i>	name	该模板适用的语言名字
<i>tpltrusted</i>	boolean	如果语言被认为是可信的则为真
<i>tpldbacreate</i>	boolean	如果语言可以被一个数据库拥有者创建则为真
<i>tplhandler</i>	text	调用处理函数的名字
<i>tplinline</i>	text	匿名阻塞处理函数的名字，如果没有则为空
<i>tplvalidator</i>	text	验证函数的名字，如果没有则为空
<i>tpllibrary</i>	text	实现语言的共享库的路径
<i>tplacl</i>	aclitem[]	模板的访问权限（并未真正使用）

目前任何命令都不能操纵过程语言模板。要改变内建信息，超级用户必须使用普通的**INSERT**、**DELETE**或**UPDATE**命令修改该表。

1.38. ux_policy

目录ux_policy存储着表的行级安全性策略。一个策略包括它适用于的命令种类（可能适用于所有命令）、它适用于的角色、被作为安全屏障条件增加到包括该表的查询的表达式以及被作为**WITH CHECK**选项增加到尝试向表增加新纪录的查询的表达式。

表 1.38. ux_policy列

名称	类型	引用	描述
<i>polname</i>	name		策略的名称
<i>polrelid</i>	oid	ux_class.oid	策略适用的表
<i>polcmd</i>	char		策略适用的命令类型： r表示 SELECT ，a表 示 INSERT ，w表 示 UPDATE ，d表

名称	类型	引用	描述
			示DELETE， *表示所有命令类型
<i>polpermissive</i>	boolean		策略是宽容性的还是限制性的？
<i>polroles</i>	oid[]	ux_authid.oid	策略适用的角色
<i>polqual</i>	ux_node_tree		被作为安全屏障条件增加到使用该表的查询的表达式树
<i>polwithcheck</i>	ux_node_tree		被作为WITH CHECK 条件增加到尝试向表增加的查询的表达式树

注意

存储在ux_policy中的策略只有在它们所适用的表的ux_class.rerowsecurity被设置时才其作用。

1. 39. ux_proc

目录ux_proc存放有关函数、过程、聚集函数以及窗口函数（共称为例程）的信息。

如果proisagg为真，则该项是一个聚集函数，在ux_aggregate中应该有一个相匹配的行。

表 1. 39. ux_proc的列

名称	类型	引用	描述
<i>oid</i>	oid		行标识符
<i>proname</i>	name		函数的名字
<i>pronamespace</i>	oid	ux_namespace.oid	函数所属的名字空间的OID
<i>proowner</i>	oid	ux_authid.oid	函数的拥有者
<i>prolang</i>	oid	ux_language.oid	实现语言或该函数的调用接口
<i>procost</i>	float4		估计的执行代价（以cpu_operator_cost为单位），如果proretset为真，这是每行返回的代价
<i>prorows</i>	float4		估计的结果行数量（如果proretset为假，该值为0）
<i>provariadic</i>	oid	ux_type.oid	可变数组参数的元素的数据类型，如果函数没有可变参数则为0
<i>prosupport</i>	regproc	ux_proc.oid	对于该函数可选的计划器支持函数

名称	类型	引用	描述
<i>prokind</i>	char		f表示普通函数，p表示过程，a表示聚集函数，w表示窗口函数
<i>prosecdef</i>	bool		函数是一个安全性定义者（即，一个“setuid”函数）
<i>proleakproof</i>	bool		该函数没有副作用。除了通过返回值，没有关于参数的信息被传播。任何会抛出基于其参数值的错误信息的函数都不是泄露验证的。
<i>proisstrict</i>	bool		当任意调用函数为空时，函数是否会返回空值。在那种情况下函数实际上根本不会被调用。非“strict”函数必须准备好处理空值输入。
<i>proretset</i>	bool		函数是否返回一个集合（即，指定数据类型的多个值）
<i>provolatile</i>	char		<i>provolatile</i> 说明函数是仅仅只依赖于它的输入参数，还是会被外部因素影响。值i表示“不变的”函数，它对于相同的输入总是输出相同的结果。值s表示“稳定的”函数，它的结果（对于固定输入）在一次扫描内不会变化。值v表示“不稳定的”函数，它的结果在任何时候都可能变化（使用v页表示函数具有副作用，所以对它们的调用无法得到优化）
<i>proparallel</i>	char		<i>proparallel</i> 说明该函数在并行模式下是否能安全地运行。对于能在并行模式下不受限制安全运行的函数，这列是s。对于可以在并行模式下运行但是只限于由并行分组的领导者执行的函数，这列是r。对于在并行模式中不安全的函数，这列是u，

名称	类型	引用	描述
			这种函数的存在会强制一个顺序执行计划。
<i>pronargs</i>	int2		输入参数的个数
<i>pronargdefaults</i>	int2		具有默认值的参数个数
<i>prorettytype</i>	oid	ux_type.oid	返回值的数据类型
<i>proargtypes</i>	oidvector	ux_type.oid	一个函数参数的数据类型 的数组。这只包括输入参数 (含INOUT和VARIADIC参 数)，因此也表现了函 数的调用特征。
<i>proallargtypes</i>	oid[]	ux_type.oid	一个函数参数的数据类型 的数组。这包括所有参 数 (含OUT和INOUT参 数)。但是，如果所有 参数都是IN参数，这 个域将为空。注意下标 是从1开始，然而由于 历史原因 <i>proargtypes</i> 的 下标是从0开始。
<i>proargmodes</i>	char[]		一个函数参数的模式的 数组。编码为： i表 示IN参数， o表 示OUT参数， b表 示INOUT参数， v表 示VARIADIC参数， t表示TABLE参数。如 果所有的参数都是 IN参数，这个域为 空。注意这里的下标对 应着 <i>proallargtypes</i> 而不 是 <i>proargtypes</i> 中的位 置。
<i>proargnames</i>	text[]		一个函数参数的名字的 数组。没有名字的参数 在数组中设置为空字符 串。如果没有一个参数 有名字，这个域为空。 注意这里的下标对应 着 <i>proallargtypes</i> 而不 是 <i>proargtypes</i> 中的位 置。
<i>proargdefaults</i>	ux_node_tree		默认值的表达式树（按 照nodeToString()的表 现方式）。这是一个 <i>pronargdefaults</i> 元素 的列表，对应于最

名称	类型	引用	描述
			后 N 个input参数（即最后 N 个 $proargtypes$ 位置）。如果没有一个参数具有默认值，这个域为空。
<i>protrftypes</i>	oid[]		要在其上应用转换的数据类型的 OID。
<i>prosrc</i>	text		这个域告诉函数处理者如何调用该函数。它可能是针对解释型语言的真实源码、一个符号链接、一个文件名或任何其他东西，这取决于实现语言/调用规范。
<i>probin</i>	text		关于如何调用函数的附加信息。其解释是与语言相关的。
<i>proconfig</i>	text[]		函数对于运行时配置变量的本地设置值
<i>proacl</i>	aclitem[]		访问权限

对于编译好的函数，包括内建的和动态载入的，*prosrc*包含了函数的C语言名字（链接符号）。对于所有其他已知的语言类型，*prosrc*包含函数的源码文本。除了对于动态载入的C函数之外，*probin*是不被使用的。对于动态载入的C函数，它给定了包含该函数的共享库文件的名称。

1.40. ux_publication

目录ux_publication包含数据库中创建的所有publication。

表 1.40. ux_publication的列

名称	类型	引用	描述
<i>oid</i>	oid		行标识符
<i>pubname</i>	name		publication的名称
<i>pubowner</i>	oid	ux_authid.oid	publication的拥有者
<i>puballtables</i>	bool		如果为真，这个publication自动包括数据库中的所有表，包括未来将会创建的任何表。
<i>pubinsert</i>	bool		如果为真，为publication中的表复制INSERT操作。
<i>pubupdate</i>	bool		如果为真，为publication中的表复制UPDATE操作。

名称	类型	引用	描述
<i>pubdelete</i>	bool		如果为真，为publication中的表复制DELETE操作。
<i>pubtruncate</i>	bool		如果为真，为publication中的表复制TRUNCATE操作。

1.41. ux_publication_rel

目录ux_publication_rel包含数据库中关系和publication之间的映射。这是一种多对多映射。这些信息对用户更加友好的视图请参考[第 1.79 节 “ux_publication_tables”](#)。

表 1.41. ux_publication_rel列

名称	类型	引用	描述
<i>prpubid</i>	oid	ux_publication.oid	对publication的引用
<i>prrelid</i>	oid	ux_class.oid	对关系的引用

1.42. ux_range

目录ux_range存储关于范围类型的信息。它是类型在[ux_type](#)中项的补充。

表 1.42. ux_range的列

名称	类型	引用	描述
<i>rngtypid</i>	oid	ux_type.oid	范围类型的OID
<i>rngsubtype</i>	oid	ux_type.oid	该范围类型的元素类型（子类型）的OID
<i>rngcollation</i>	oid	ux_collation.oid	用于范围比较的排序规则的OID，如果没有则为0
<i>rngsubopc</i>	oid	ux_opclass.oid	用于范围比较的子类型的操作符类的OID
<i>rngcanonical</i>	regproc	ux_proc.oid	将一个范围值转换为规范形式的函数的OID，如果没有则为0
<i>rngsubdiff</i>	regproc	ux_proc.oid	以双精度返回两个元素值不同的函数的OID，如果没有则为0

rngsubopc（加上*rngcollation*，如果元素类型是可排序的）决定了被该范围类型所使用的排序顺序。*rngcanonical*用于离散类型的元素类型。*rngsubdiff*是可选的，但是提供它可以提高范围类型上的GiST索引性能。

1.43. ux_replication_origin

ux_replication_origin目录包含所有已创建的复制源。

和大部分系统目录不同，`ux_replication_origin`在一个集簇的所有数据库之间共享：每个集簇只有一份`ux_replication_origin`拷贝，而不是每个数据库一份。

表 1.43. `ux_replication_origin`的列

名称	类型	描述
<i>roident</i>	oid	一个集簇范围内唯一的复制源标识符。应该绝不会脱离系统。
<i>roname</i>	text	外部的由用户定义的复制源名称。

1.44. `ux_rewrite`

目录`ux_rewrite`存储对于表和视图的重写规则。

表 1.44. `ux_rewrite`的列

名称	类型	引用	描述
<i>oid</i>	oid		行标识符
<i>rulename</i>	name		规则名称
<i>ev_class</i>	oid	ux_class.oid	使用该规则的表
<i>ev_type</i>	char		使用该规则的事件类型：1 = SELECT, 2 = UPDATE, 3 = INSERT, 4 = DELETE
<i>ev_enabled</i>	char		控制在哪种 <code>session_replication_role</code> 模式中触发该规则。 O = 规则在“origin”和“local”模式触发，D = 规则被禁用，R = 规则在“replica”模式触发，A = 规则总是被触发。
<i>is_instead</i>	bool		为真表示是一个INSTEAD规则
<i>ev_qual</i>	ux_node_tree		规则条件的表达式树（按照 <code>nodeToString()</code> 的表现形式）
<i>ev_action</i>	ux_node_tree		规则动作的查询树（按照 <code>nodeToString()</code> 的表现形式）

注意

如果一个表在这个目录中有任何规则，`ux_class.relhasrules`必须为真。

1.45. ux_seclabel

目录ux_seclabel存储数据库对象上的安全标签。安全标签可以通过SECURITY LABEL命令操纵。简单的查看安全标签方法请见[第 1.84 节 “ux_seclabels”](#)。

同时请见[ux_shseclabel](#)，它对集簇共享的数据库对象的安全标签执行相似的功能。

表 1.45. ux_seclabel的列

名称	类型	引用	描述
<i>objoid</i>	oid	任意OID列	该安全标签依附的对象的OID
<i>classoid</i>	oid	ux_class.oid	该对象所出现的系统目录的OID
<i>objsubid</i>	int4		对于一个在表列上的安全标签，这将是列号（ <i>objoid</i> 和 <i>classoid</i> 指表本身）。对于所有其他对象类型，本列为0。
<i>provider</i>	text		与该标签相关的标签提供者。
<i>label</i>	text		应用于该对象的安全标签。

1.46. ux_sequence

目录ux_sequence包含有关序列的信息。一些序列的信息（例如名称和方案）放在ux_class中。

表 1.46. ux_sequence的列

名称	类型	引用	描述
<i>seqrelid</i>	oid	ux_class.oid	这个序列的ux_class项的OID
<i>seqtypid</i>	oid	ux_type.oid	序列的数据类型
<i>seqstart</i>	int8		序列的起始值
<i>seqincrement</i>	int8		序列的增量值
<i>seqmax</i>	int8		序列的最大值
<i>seqmin</i>	int8		序列的最小值
<i>seqcache</i>	int8		序列的缓冲尺寸
<i>seqcycle</i>	bool		序列是否循环

1.47. ux_shdepend

目录ux_shdepend记录数据库对象和共享对象之间的依赖关系，例如角色。这些信息使得UXsinoDB可以确保对象在被删除时没有被其他对象引用。

另请参阅[ux_depend](#)，它对单个数据库中对象之间的依赖提供了相似的功能。

与大部分其他系统目录不同，`ux_shdepend`在整个集簇的所有数据库之间共享：在每一个集簇中只有一个`ux_shdepend`的拷贝，而不是每个数据库一份。

表 1.47. `ux_shdepend`的列

名称	类型	引用	描述
<i>dbid</i>	oid	ux_database.oid	依赖者对象所在的数据库OID，如果是一个共享对象则值为0
<i>classid</i>	oid	ux_class.oid	依赖者对象所在的系统目录的OID
<i>objid</i>	oid	任意OID列	依赖者对象的OID
<i>objsubid</i>	int4		对于一个表列，这将是列号（ <i>objid</i> 和 <i>classid</i> 指向表本身）。对于所有其他对象类型，该列值为0。
<i>refclassid</i>	oid	ux_class.oid	被引用对象所在的系统目录的OID（必须是一个共享的目录）
<i>refobjid</i>	oid	任意OID列	被引用对象的OID
<i>deptype</i>	char		定义该依赖关系的特定语义的代码，见表后的说明

在所有情况下，一个`ux_shdepend`项表明被引用对象不能在没有删除其依赖对象的情况下被删除。但是，其中也有多种依赖类型，由*deptype*标识：

SHARED_DEPENDENCY_OWNER (o)

被引用对象（必须是一个角色）是依赖对象的拥有者。

SHARED_DEPENDENCY_ACL (a)

被引用对象（必须是一个角色）在依赖对象的ACL（访问控制列表，即权限列表）中被提到。（不会为对象的拥有者创建一个SHARED_DEPENDENCY_ACL项，因为拥有者将会有有一个SHARED_DEPENDENCY_OWNER项。）

SHARED_DEPENDENCY_POLICY (r)

作为一个依赖策略对象的目标被引用的对象（必须是一个角色）。

SHARED_DEPENDENCY_PIN (p)

没有依赖对象，这种类型的项是一个信号，用来指示系统本身依赖于被引用对象，并且因此该对象必须永远不能被删除。这种类型的项只能被`initdb`创建。这种项中关于依赖对象的列值都为0。

未来可能会需要其他的依赖类型。特别要注意的是在当前定义中只支持角色作为被引用对象。

1.48. `ux_shdescription`

目录`ux_shdescription`存储共享数据库对象的可选描述（注释）。描述可以通过`COMMENT`命令操作，并且可以使用`uxsql`的`\d`命令来查看。

另请参阅[ux_description](#)，它对单个数据库中对象之间的依赖提供了相似的功能。

与大部分其他系统目录不同，`ux_shdescription`在整个集簇的所有数据库之间共享：在每一个集簇中只有一个`ux_shdescription`的拷贝，而不是每个数据库一份。

表 1.48. `ux_shdescription`的列

名称	类型	引用	描述
<i>objoid</i>	oid	任意OID列	该描述所属的对象的OID
<i>classoid</i>	oid	ux_class.oid	该对象所在系统目录的OID
<i>description</i>	text		作为该对象描述的任何文本

1.49. `ux_shseclabel`

目录`ux_shseclabel`存储共享数据库对象上的安全标签。安全标签可以通过`SECURITY LABEL`命令操纵。更简单的查看安全标签的方式请见[第 1.84 节 “ux_seclabels”](#)

另请参阅[ux_seclabel](#)，它对单个数据库中对象的安全标签提供了相似的功能。

与大部分其他系统目录不同，`ux_shseclabel`在整个集簇的所有数据库之间共享：在每一个集簇中只有一个`ux_shseclabel`的拷贝，而不是每个数据库一份。

表 1.49. `ux_shseclabel`的列

名称	类型	引用	描述
<i>objoid</i>	oid	任意OID列	该安全标签所属对象的OID
<i>classoid</i>	oid	ux_class.oid	对象所属系统目录的OID
<i>provider</i>	text		与此标签关联的标签提供者
<i>label</i>	text		应用到该对象的安全标签

1.50. `ux_statistic`

目录`ux_statistic`存储有关数据库内容的统计数据。其中的项由`ANALYZE`创建，查询规划器会使用这些数据来进行查询规划。注意所有的统计数据天然就是近似的，即使它刚刚被更新。

通常对于数据表中一个已经被 `ANALYZE` 过的列，在本目录中会存在一个`stainherit = false`的项。如果该列所在的表具有后代（即有其他表继承该表），对于该列还会创建第二个`stainherit = true`的项。`stainherit = true`的项表示列在整个继承树上的统计数据，即通过`SELECT column FROM table*`看到的数据的统计，而`stainherit = false`的项表示对`SELECT column FROM ONLY table`的结果的统计。

`ux_statistic`也存储关于索引表达式值的统计数据，就好像它们是真正的数据列，但在这种情况下`starelid`指索引。对一个普通非表达式索引列不会创建项，因为它将是底层表列的项的冗余。当前，索引表达式的项都具有`stainherit = false`。

因为不同类型的统计信息适用于不同类型的数据，`ux_statistic`被设计成不太在意自己存储的是什么类型的统计。只有极为常用的统计信息（比如NULL的含量）才在`ux_statistic`里给予专用的字段。其它所有东西都存储在“槽位”中，而槽位是一组相关的列，它们的内容用槽位中的一个列里的代码表示。更详细的信息请参阅 `src/include/catalog/ux_statistic.h`。

`ux_statistic`不应该是公共可读的，因为即使是一个表内容的统计性信息也可能被认为是敏感的（例子：一个薪水列的最大和最小值可能是非常有趣的）。`ux_stats`是`ux_statistic`上的一个公共可读的视图，它只会显示出当前用户可读的表的信息。

表 1.50. `ux_statistic`的列

名称	类型	引用	描述
<i>starelid</i>	oid	ux_class.oid	被描述列所属的表或索引
<i>staattnum</i>	int2	ux_attribute.attnum	被描述列的编号
<i>stainherit</i>	bool		如果为真，统计包含了继承后代的列而不仅仅是指定关系的列
<i>stanullfrac</i>	float4		列的项为空的比例
<i>stawidth</i>	int4		非空项的平均存储宽度，以字节计
<i>stadistinct</i>	float4		列中非空唯一值的数目。一个大于零的值是唯一值的真正数目。一个小于零的值是表中行数的乘数的负值；例如，对于一个 80% 的值为非空且每个非空值平均出现两次的列，可以表示为 <code>stadistinct = -0.4</code> 。一个0值表示唯一值的数目未知。
<i>stakindN</i>	int2		一个代码，它表示存储在该 <code>ux_statistic</code> 行中第N个“槽位”的统计类型。
<i>staopN</i>	oid	ux_operator.oid	一个用于生成这些存储在第N个“槽位”的统计信息的操作符。比如，一个柱面图槽位会用<操作符，该操作符定义了该数据的排序顺序。
<i>stacollN</i>	oid	ux_collation.oid	排序规则用于导出存储在第N个“槽”中的统计信息。例如，可排序列的直方图槽将显示定

名称	类型	引用	描述
			义数据排序顺序的排序规则。对于不可整理数据，为零。
<i>stanumbersN</i>	float4[]		第N个“槽位”的类型的数值类型统计，如果该槽位不涉及数值类型则为NULL
<i>stavaluesN</i>	anyarray		第N个“槽位”的类型的列值，如果该槽位类型不存储任何数据值则为 NULL。每个数组的元素值实际上都是指定列的数据类型或者是一个相关类型（如数组元素类型），因此，除了把这些列的类型定义成anyarray之外别无他法。

1.51. ux_statistic_ext

目录ux_statistic_ext包含了扩展的规划器统计信息的定义。这个目录中的每一行对应于一个用CREATE STATISTICS创建的统计信息对象。

表 1.51. ux_statistic_ext的列

名称	类型	引用	描述
<i>stxrelid</i>	oid	ux_class.oid	包含这个对象所描述的列的表
<i>stxname</i>	name		统计信息对象的名称
<i>stxnamespace</i>	oid	ux_namespace.oid	包含这个统计信息对象的名字空间的OID
<i>stxowner</i>	oid	ux_authid.oid	统计信息对象的拥有者
<i>stxkeys</i>	int2vector	ux_attribute.attnum	一个属性编号的数组，表示哪些表列被这个统计信息对象覆盖。例如值1 3表示第一个和第三个表列被覆盖
<i>stxkind</i>	char[]		包含被启用统计类型代码的数组，可用的值有： d 表示n-distinct统计信息， f 表示函数依赖统计信息和 m 表示最常见值（MCV）列表的统计信息

ux_statistic_ext条目在CREATE STATISTICS期间完全填充，但是随后不计算实际的统计值。后来ANALYZE命令计算所需的值，并在[ux_statistic_ext_data](#)目录中填充条目。

1.52. ux_statistic_ext_data

目录ux_statistic_ext_data保存在ux_statistic_ext中定义的扩展规划器统计信息的数据。该目录的每一行对应用CREATE STATISTICS创建的一个统计信息对象。

就像ux_statistic，ux_statistic_ext_data不应该被公众读取到，因为其内容可能被视为敏感内容。（例如：列中值的最常见组合可能非常令人关注。）[ux_stats_ext](#)是ux_statistic_ext_data上的公共可读视图（与ux_statistic_ext连接后），它仅暴露有关当前用户可读的表和列的信息。

表 1.52. ux_statistic_ext_data Columns

名称	类型	引用	描述
<i>stxoid</i>	oid	ux_statistic_ext.oid	包含此数据定义的扩展统计信息对象
<i>stxdndistinct</i>	ux_ndistinct		N-distinct计数，序列化为ux_ndistinct类型
<i>stxddependencies</i>	ux_dependencies		函数依赖统计信息，序列化为ux_dependencies类型
<i>stxdmcv</i>	ux_mcv_list		MCV（最频值）列表统计信息，序列化为ux_mcv_list类型

1.53. ux_subscription

目录ux_subscription包含所有现有的逻辑复制订阅。

和大部分系统目录不同，ux_subscription在集簇的所有数据库之间共享：每个集簇只有一份ux_subscription拷贝，而不是每个数据库一份。

对列subconninfo的访问被从普通用户那里收回，因为该列可能含有明文口令。

表 1.53. ux_subscription的列

名称	类型	引用	描述
<i>oid</i>	oid		行标识符
<i>subdbid</i>	oid	ux_database.oid	订阅所在的数据库的OID
<i>subname</i>	name		订阅的名称
<i>subowner</i>	oid	ux_authid.oid	订阅的拥有者
<i>subenabled</i>	bool		如果为真，订阅被启用并且应该被复制。
<i>subsynccommit</i>	text		包含订阅工作者的synchronous_commit设置的值。
<i>subconninfo</i>	text		到上游数据库的连接字符串

名称	类型	引用	描述
<i>subslotname</i>	name		上游数据库中的复制槽的名称。也被用于本地复制源名称。
<i>subpublications</i>	text[]		被订阅的publication名称的数组。这些引用的是发布者服务器上的publication。

1.54. ux_subscription_rel

目录ux_subscription_rel包含每个订阅中每个被复制关系的状态。这是一种多对多映射。

这个目录仅包含运行CREATE SUBSCRIPTION或ALTER SUBSCRIPTION ... REFRESH PUBLICATION以后对订阅已知的表。

表 1.54. ux_subscription_rel的列

名称	类型	引用	描述
<i>srsubid</i>	oid	ux_subscription.oid	对订阅的引用
<i>srrelid</i>	oid	ux_class.oid	对关系的引用
<i>srsubstate</i>	char		状态代码： i = 初始化， d = 数据正在被拷贝， s = 已同步， r = 准备好（普通复制）
<i>srsublsn</i>	ux_lsn		s和r状态的结束LSN。

1.55. ux_tablespace

目录ux_tablespace存储关于可用表空间的信息。表可以被放置在特定表空间中以实现磁盘布局的管理。

与大部分其他系统目录不同，ux_tablespace在整个集簇的所有数据库之间共享：在每一个集簇中只有一个ux_tablespace的拷贝，而不是每个数据库一份。

表 1.55. ux_tablespace的列

名称	类型	引用	描述
<i>oid</i>	oid		行标识符
<i>spcname</i>	name		表空间名
<i>spcowner</i>	oid	ux_authid.oid	表空间的拥有者，通常是创建它的用户
<i>spcacl</i>	aclitem[]		访问权限
<i>spcoptions</i>	text[]		表空间级别的选项，形如“keyword=value”的字符串

1.56. ux_transform

目录ux_transform存储有关转换的信息，转换是一种让数据类型适应过程语言的机制。

表 1.56. ux_transform的列

名称	类型	引用	描述
<i>trftype</i>	oid	ux_type.oid	这个转换所针对的数据类型的 OID
<i>trflang</i>	oid	ux_language.oid	这个转换所针对的语言的 OID
<i>trffromsql</i>	regproc	ux_proc.oid	一个函数的 OID，该函数用来将数据类型转换为过程语言的输入（例如 函数参数）。如果不支持这种操作，这里存储零。
<i>trftosql</i>	regproc	ux_proc.oid	一个函数的 OID，该函数被用来转换过程语言的输出（例如返回值）为该数据类型。如果不支持这种操作，这里存储零。

1.57. ux_trigger

目录ux_trigger存储表和视图上的触发器。

表 1.57. ux_trigger的列

名称	类型	引用	描述
<i>oid</i>	oid		行标识符
<i>tgrelid</i>	oid	ux_class.oid	触发器所在的表
<i>tgname</i>	name		触发器名（在同一个表的触发器中必须唯一）
<i>tgfoid</i>	oid	ux_proc.oid	要被触发器调用的函数
<i>tgtype</i>	int2		触发器触发条件的位掩码
<i>tgenabled</i>	char		控制触发器在 session_replication_role 模式中的触发。 O = 触发器在“origin”和“local”模式触发， D = 触发器被禁用， R = 触发器在“replica”模式触发， A = 触发器总是触发。

名称	类型	引用	描述
<i>tgisinternal</i>	bool		为真表示触发器是内部生成的（通常是为了强制由 <i>tgconstraint</i> 指定的约束）
<i>tgconstrrelid</i>	oid	ux_class.oid	被一个引用完整性约束引用的表
<i>tgconstrindid</i>	oid	ux_class.oid	支持一个唯一、主键、引用完整性约束或者排除约束的索引
<i>tgconstraint</i>	oid	ux_constraint.oid	可能存在的与触发器相关的ux_constraint项
<i>tgdeferrable</i>	bool		如果约束触发器可推迟则为真
<i>tginitdeferred</i>	bool		如果约束触发器初始可推迟则为真
<i>tgargs</i>	int2		传递给触发器函数的参数字符串个数
<i>tgattr</i>	int2vector	ux_attribute.attnum	如果触发器是列限定的，这里存放列号；否则这是一个空数组
<i>tgargs</i>	bytea		传递给触发器的参数字符串，每一个都以NULL结尾
<i>tgqual</i>	ux_node_tree		触发器WHEN条件的表达式树（以nodeToString()的表现形式），如果没有则为空
<i>tgoldtable</i>	name		OLD TABLE的REFERENCING子句名称，如果没有则为空
<i>tgnewtable</i>	name		NEW TABLE的REFERENCING子句名称，如果没有则为空

当前，列限定触发器只被UPDATE事件支持，因此*tgattr*只用于这种事件类型。*tgtype*页可以包含用于其他事件类型的位，但其他事件类型是表范围的触发器且会忽略*tgattr*。

注意

当*tgconstraint*非零时，*tgconstrrelid*、*tgconstrindid*、*tgdeferrable*和*tginitdeferred*与被引用的ux_constraint项有很大的冗余。但是，存在将一个不可延迟触发器关联到一个可延迟约束的可能性：外键约束可以有一些可延迟和一些不可延迟触发器。

注意

如果一个关系在本目录中拥有任何触发器，其`ux_class.relhastriggers`必须为真。

1.58. ux_ts_config

`ux_ts_config`目录包含表示文本搜索配置的选项。一个配置指定了一个特定的文本搜索分析器和一个用于分析器输出记号的字典列表。分析器由`ux_ts_config`项展现，而记号到字典的映射则由[ux_ts_config_map](#)中的辅助项定义。

表 1.58. `ux_ts_config`的列

名称	类型	引用	描述
<i>oid</i>	oid		行标识符
<i>cfgname</i>	name		文本搜索配置名
<i>cfgnamespace</i>	oid	ux_namespace.oid	包含该配置的名字空间的OID
<i>cfgowner</i>	oid	ux_authid.oid	配置的拥有者
<i>cfgparser</i>	oid	ux_ts_parser.oid	该配置的文本搜索分析器的OID

1.59. ux_ts_config_map

`ux_ts_config_map`目录包含的项展示了对于每一个文本搜索配置的每一种输出记号类型，有哪些文本搜索字典可供查询以及以何种顺序。

表 1.59. `ux_ts_config_map`的列

名称	类型	引用	描述
<i>mapcfg</i>	oid	ux_ts_config.oid	拥有该映射项的 <code>ux_ts_config</code> 项的OID
<i>maptokentype</i>	integer		一种由配置的分析器送出的记号类型
<i>mapseqno</i>	integer		查询该项的顺序（ <i>mapseqno</i> 值小的优先）
<i>mapdict</i>	oid	ux_ts_dict.oid	查询的文本搜索字典的OID

1.60. ux_ts_dict

`ux_ts_dict`目录包含定义文本搜索字典的项。一个字典依赖于一个文本搜索模板，它指定了所有需要的函数实现，字典本身则为模板支持的用户可设置参数提供值。这种分工允许无权限的用户创建字典。参数由一个文本串`dictinitoption`定义，其格式和意义随着模板而变化。

表 1.60. ux_ts_dict的列

名称	类型	引用	描述
<i>oid</i>	oid		行标识符
<i>dictname</i>	name		文本搜索字典名
<i>dictnamespace</i>	oid	ux_namespace.oid	包含该字典的名字空间OID
<i>dictowner</i>	oid	ux_authid.oid	字典的拥有者
<i>dicttemplate</i>	oid	ux_ts_template.oid	该字典的文本搜索模板的OID
<i>dictinitoption</i>	text		模板的初始化选项串

1.61. ux_ts_parser

ux_ts_parser目录包含定义文本搜索分析器的项。一个分析器负责将输入文本分割成词位并为每一个词位分配一个记号类型。由于一个分析器必须用C语言级别的函数实现，创建新分析器的工作只限于数据库的超级用户。

表 1.61. ux_ts_parser的列

名称	类型	引用	描述
<i>oid</i>	oid		行标识符
<i>prsname</i>	name		文本搜索分析器的名字
<i>prsnamespace</i>	oid	ux_namespace.oid	包含此分析器的名字空间的OID
<i>prsstart</i>	regproc	ux_proc.oid	分析器启动函数的OID
<i>prstoken</i>	regproc	ux_proc.oid	分析器的下一记号函数的OID
<i>prsend</i>	regproc	ux_proc.oid	分析器的关闭函数的OID
<i>prsheadline</i>	regproc	ux_proc.oid	分析器的大标题函数的OID
<i>prsllextype</i>	regproc	ux_proc.oid	分析器的词汇类型函数的OID

1.62. ux_ts_template

ux_ts_template目录包含定义文本搜索模板的项。一个模板是一类文本搜索字典的实现骨架。由于一个模板必须用C语言级别的函数实现，新模板的创建只限于数据库超级用户。

表 1.62. ux_ts_template的列

名称	类型	引用	描述
<i>oid</i>	oid		行标识符
<i>tmplname</i>	name		文本搜索模板的名字

名称	类型	引用	描述
<i>tplnamespace</i>	oid	ux_namespace.oid	包含此模板的名字空间的OID
<i>tplinit</i>	regproc	ux_proc.oid	模板的初始化函数的OID
<i>tpllexize</i>	regproc	ux_proc.oid	模板的词汇化函数的OID

1.63. ux_type

目录ux_type存储有关数据类型的信息。基类和枚举类型（标度类型）使用CREATE TYPE创建，而域使用CREATE DOMAIN创建。数据库中的每一个表都会有一个自动创建的组类型，用于表示表的行结构。也可以使用CREATE TYPE AS创建组类型。

表 1.63. ux_type的列

名称	类型	引用	描述
<i>oid</i>	oid		行标识符
<i>typname</i>	name		数据类型的名字
<i>typnamespace</i>	oid	ux_namespace.oid	包含此类型的名字空间的OID
<i>typowner</i>	oid	ux_authid.oid	类型的拥有者
<i>typlen</i>	int2		对于一个固定尺寸的类型， <i>typlen</i> 是该类型内部表示的字节数。对于一个变长类型， <i>typlen</i> 为负值。-1表示一个“varlena”类型（具有长度字），-2表示一个以空值结尾的C字符串。
<i>typbyval</i>	bool		<i>typbyval</i> 判断内部例程传递这个类型的数值时是通过传值还是传引用。如果 <i>typlen</i> 不是1、2或4（或者在Datum为8字节的机器上为8），因此 <i>typbyval</i> 最好是假。变长类型总是传引用。注意即使长度允许传值， <i>typbyval</i> 也可以为假。
<i>typtype</i>	char		<i>typtype</i> 可以是：b表示一个基类，c表示一个组类型（例如一个表的行类型），d表示一个域，e表示一个枚举类型，p表示一

名称	类型	引用	描述
			个伪类型，或 <code>r</code> 表示一个范围类型。另请参阅 <i>typrelid</i> 和 <i>typbasetype</i> 。另请参阅 <i>typrelid</i> 和 <i>typbasetype</i> 。
<i>typcategory</i>	char		<i>typcategory</i> 是一种任意的数据类型分类，它被分析器用来决定哪种隐式转换“更好”。参见表 1.64 “ <i>typcategory</i> 编码”。
<i>typispreferred</i>	bool		如果此类型在它的 <i>typcategory</i> 中是一个更好的转换目标，此列为真
<i>typisdefined</i>	bool		如果此类型已被定义则为真，如果此类型只是一个表示还未定义类型的占位符则为假。当 <i>typisdefined</i> 为假，除了类型名字、名字空间和OID之外什么都不能被依赖。
<i>typdelim</i>	char		在分析数组输入时，分隔两个此类型值的字符。注意该分隔符是与数组元素数据类型相关联的，而不是和数组的数据类型关联。
<i>typrelid</i>	oid	ux_class.oid	如果这是一个复合类型（见 <i>typtype</i> ），那么这个列指向 <i>ux_class</i> 中定义对应表的项（对于自由存在的复合类型， <i>ux_class</i> 项并不表示一个表，但不管怎样该类型的 <i>ux_attribute</i> 项需要链接到它）。对非复合类型此列为零。
<i>typelem</i>	oid	ux_type.oid	如果 <i>typelem</i> 不为0，则它标识 <i>ux_type</i> 里面的另外一行。当前类型可以被加上下标得到一个值为类型 <i>typelem</i> 的数组来描述。一个“真的”数组类型是变长的（ <i>typlen</i> = -1），但是一些定长的（ <i>typlen</i> > 0）类型也

名称	类型	引用	描述
			拥有非零的 <code>typelem</code> ，比如 <code>name</code> 和 <code>point</code> 。如果一个定长类型拥有一个 <code>typelem</code> ，则它的内部形式必须是某个 <code>typelem</code> 数据类型的值，不能有其它数据。变长数组类型有一个由该数组子例程定义的头。
<code>typarray</code>	oid	ux_type.oid	如果 <code>typarray</code> 不是0，则它标识 <code>ux_type</code> 中的另一行，这一行是一个将此类型作为元素的“真的”数组类型
<code>typinput</code>	regproc	ux_proc.oid	输入转换函数（文本格式）
<code>typoutput</code>	regproc	ux_proc.oid	输出转换函数（文本格式）
<code>typreceive</code>	regproc	ux_proc.oid	输入转换函数（二进制格式），如果没有则为0
<code>typsend</code>	regproc	ux_proc.oid	输出转换函数（二进制格式），如果没有则为0
<code>typmodin</code>	regproc	ux_proc.oid	类型修改器输入函数，如果类型没有提供修改器则为0
<code>typmodout</code>	regproc	ux_proc.oid	类型修改器输出函数，如果类型没有提供修改器则为0
<code>typanalyze</code>	regproc	ux_proc.oid	自定义ANALYZE函数，0表示使用标准函数
<code>typalign</code>	char		<code>typalign</code> 是当存储此类类型值时要求的对齐性质。它应用于磁盘存储以及该值在 <code>UXsinoDB</code> 内部的大多数表现形式。如果数值是连续存放的，比如在磁盘上的一个完整行，在这种类型的数据前会插入填充，这样它就可以按照指定边界存储。对齐引用是该序列中第一个数据的开头。对齐引用是序列中第一个数据的开始。

名称	类型	引用	描述
			<p>可能的值有：</p> <ul style="list-style-type: none"> • c = char对齐，即不需要对齐。 • s = short对齐（在大部分机器上为2字节）。 • i = int对齐（在大部分机器上为4字节）。 • d = double对齐（在很多机器上为8字节，但绝不是全部）。 <p>注意：对于系统表中使用的类型，很关键的是，<code>ux_type</code>中定义的尺寸和对齐方式要和编译器在表示表行的结构中布局列的方式保持一致。</p>
<i>typstorage</i>	char		<p>如果一个变长类型（<i>typen</i> = -1）可被TOAST，<i>typstorage</i>说明这种类型的列应采取的默认策略。可能的值是：</p> <ul style="list-style-type: none"> • p：值必须平面存储。 • e：值可以被存储在一个“二级”关系（如果有，见<code>ux_class.reltoastrelid</code>）。 • m：值可以被压缩线内存储。 • x：值可以被压缩线内存储或存储在“二级”存储。 <p>注意m列也可以被移动到二级存储，但只能作为最后一种方案（e和x列会先被移动）。</p>

名称	类型	引用	描述
<i>typnotnull</i>	bool		<i>typnotnull</i> 表示类型上的一个非空约束。只用于域。
<i>typbasetype</i>	oid	ux_type.oid	如果这是一个域（见 <i>typtype</i> ），则 <i>typbasetype</i> 标识这个域基于的类。如果此类不是一个域则为0。
<i>typtypmod</i>	int4		域使用 <i>typtypmod</i> 来记录被应用于它们基类型的 <i>typmod</i> （如果基类型不使用 <i>typmod</i> ，则为-1）。如果此类型不是一个域则为-1。
<i>typndims</i>	int4		对于一个数组上的域， <i>typndims</i> 是数组维度数（即， <i>typbasetype</i> 是一个数组类型）。除数组类型上的域之外的类型的此列为0。
<i>typcollation</i>	oid	ux_collation.oid	<i>typcollation</i> 指定此类型的排序规则。如果类型不支持排序规则，此列为0。一个支持排序规则的基类型此处会有一个非零值，典型值为DEFAULT_COLLATION_OID。可排序类型上的域可以有一个不同于其基类型的排序规则OID，如果为该域指定了一个排序规则OID的话。
<i>typdefaultbin</i>	ux_node_tree		如果 <i>typdefaultbin</i> 为非空，那么它是该类型默认表达式的nodeToString()表现形式。这个列只用于域。
<i>typdefault</i>	text		如果某类型没有相关默认值，那么 <i>typdefault</i> 为空。如果 <i>typdefaultbin</i> 不为空，那么 <i>typdefault</i> 必须包含一个 <i>typdefaultbin</i> 所指的默认表达式的人类可读的版本。如果 <i>typdefaultbin</i> 为空

名称	类型	引用	描述
			但 <code>typdefault</code> 不为空，则 <code>typdefault</code> 是该类型默认值的外部表现形式，它可以被交给该类型的输入转换器来产生一个常量。
<code>typacl</code>	<code>aclitem[]</code>		访问权限

表 1.64 [“`typcategory`编码”](#) 列出了`typcategory`的系统定义值。任何未来对此列表的增加都将是 大写ASCII字母。所有其他ASCII字符都保留给用户定义类别。

表 1.64. `typcategory`编码

编码	类别
A	数组类型
B	布尔类型
C	组合类型
D	日期/时间类型
E	枚举类型
G	几何类型
I	网络地址类型
N	数字类型
P	伪类型
R	范围类型
S	字符串类型
T	时间间隔类型
U	用户定义类型
V	位串类型
X	未知类型

1.64. `ux_user_mapping`

目录`ux_user_mapping`存储从本地用户到远程的映射。对这个目录的访问对普通用户有限制，可使用视图[`ux_user_mappings`](#)替代。

表 1.65. `ux_user_mapping`的列

名称	类型	引用	描述
<code>oid</code>	<code>oid</code>		行标识符
<code>umuser</code>	<code>oid</code>	<code>ux_authid.oid</code>	将要被映射的本地角色的OID，如果用户映射是公共的则为0
<code>umserver</code>	<code>oid</code>	<code>ux_foreign_server.oid</code>	包含此映射的外部服务器的OID

名称	类型	引用	描述
<i>umoptions</i>	text[]		用户映射相关选项，以“keyword=value”字符串形式

1.65. 系统视图

除系统目录外，UXsinoDB提供了一些内建视图。一些系统视图为系统目录上一些常用查询提供了便利的访问。其他视图提供了对内部服务器状态的访问。

信息模式提供了一组可供选择的视图，它和系统视图在功能上有所重叠。由于信息模式是SQL标准，而这里描述的视图是UXsinoDB特有的，如果信息模式能提供你所需要的信息，通常最好使用它。

表 1.66 “系统视图”列出了这里描述的系统视图。

除了特别注明的，所有这里描述的视图都是只读的。

表 1.66. 系统视图

视图名字	用途
ux_available_extensions	可用的扩展
ux_available_extension_versions	所有版本的扩展
ux_config	编译时配置参数
ux_cursors	打开的游标
ux_file_settings	配置文件内容摘要
ux_group	数据库用户组
ux_hba_file_rules	客户端认证配置文件内容的摘要
ux_indexes	索引
ux_locks	当前保持或者等待的锁
ux_matviews	物化视图
ux_policies	策略
ux_prepared_statements	预备好的语句
ux_prepared_xacts	预备好的事务
ux_publication_tables	publication和它们相关的表
ux_replication_origin_status	有关复制源的信息，包括复制进度
ux_replication_slots	复制槽信息
ux_roles	数据库角色
ux_rules	规则
ux_seclabels	安全标签
ux_sequences	序列
ux_settings	参数设置
ux_shadow	数据库用户
ux_stats	规划器统计信息

视图名字	用途
ux_stats_ext	扩展的计划器统计信息
ux_tables	表
ux_timezone_abbrevs	时区简写
ux_timezone_names	时区名字
ux_user	数据库用户
ux_user_mappings	用户映射
ux_views	视图

1.66. `ux_available_extensions`

`ux_available_extensions`视图列出了可用于安装的扩展。参见[ux_extension](#)目录，它显示当前已安装的扩展。

表 1.67. `ux_available_extensions`的列

名字	类型	描述
<i>name</i>	name	扩展名
<i>default_version</i>	text	默认版本的名称，如果没有指定则为NULL
<i>installed_version</i>	text	当前已安装的扩展版本，如果没有安装则为NULL
<i>comment</i>	text	来自于扩展的控制文件的注释字符串

`ux_available_extensions`视图是只读的。

1.67. `ux_available_extension_versions`

`ux_available_extension_versions`视图列出了可用于安装的指定扩展版本。参见[ux_extension](#)目录，它显示当前已安装的扩展。

表 1.68. `ux_available_extension_versions`的列

名字	类型	描述
<i>name</i>	name	扩展名
<i>version</i>	text	版本名
<i>installed</i>	bool	如果此版本的扩展当前已安装则为真
<i>superuser</i>	bool	如果只有超级用户被允许安装此扩展则为真
<i>relocatable</i>	bool	如果扩展能被重定位到另一个模式则为真
<i>schema</i>	name	此扩展必须被安装到的模式名，如果此扩展是部分或者全

名字	类型	描述
		部可以重定位的，此列为NULL
<i>requires</i>	name[]	先决条件扩展的名字，如果没有则为NULL
<i>comment</i>	text	来自于扩展的控制文件的注释字符串

ux_available_extension_versions视图是只读的。

1.68. ux_config

视图ux_config描述了当前安装的UXsinoDB版本中的编译时配置参数。它存在的本意是用于那些要和UXsinoDB交互的软件包，让它们能找到所需要的头文件和库。它提供了和ux_config UXsinoDB客户端应用相同的基本信息。

默认情况下，ux_config视图只能由超级用户读取。

表 1.69. ux_config列

名称	类型	描述
<i>name</i>	text	参数名
<i>setting</i>	text	参数值

1.69. ux_cursors

ux_cursors视图列出了当前可用的游标。游标可以以几种方式定义：

- 通过SQL中的DECLARE语句
- 通过前端/后端协议中的绑定消息
- 通过服务器编程接口（SPI）

ux_cursors视图显示由任何这些方式创建的游标。视图只存在于定义它们的事务期间，除非声明了WITH HOLD。因此非保持游标只在它们的创建事务结束前存在于这个视图中。

注意

视图用于在内部实现UXsinoDB的某些部件，例如过程语言。因此，ux_cursors视图可能包括那些不是由用户显式创建的游标。

表 1.70. ux_cursors的列

名字	类型	描述
<i>name</i>	text	游标名
<i>statement</i>	text	提交用于定义此游标的查询语句

名字	类型	描述
<i>is_holdable</i>	boolean	如果游标是可保持的（即，它可以在其定义事务提交后被访问）则为true，否则为否
<i>is_binary</i>	boolean	如果游标被声明为BINARY则为true，否则为false
<i>is_scrollable</i>	boolean	如果游标是可滚动的（即，允许以一种非顺序的方式检索行）则为true，否则为false
<i>creation_time</i>	timestampz	游标被声明的时间

ux_cursors视图是只读的。

1.70. ux_file_settings

视图ux_file_settings提供了服务器配置文件 内容的概要。这个视图中的每一行表示配置文件中出现的一个 “name = value” 项，还带有注解指示该值是否被成功地应用。在 配置文件有问题时，有可能出现额外的行，它们没有相关的 “name = value” 项，一个例子是配置文件中语法错误。

这个视图有助于检查在配置文件中打算做的修改是否能工作，或者用来诊断 之前的失败。注意这个视图报告的是配置文件的当前内容， 而不是服务器最后应用的值（这些值通常查看 [ux_settings](#) 视图就够了）。

默认情况下，ux_file_settings视图只有超级用户可读。

表 1.71. ux_file_settings的列

名称	类型	描述
<i>sourcefile</i>	<i>text</i>	配置文件的完整路径名
<i>sourceline</i>	<i>integer</i>	该项在配置文件中出现的行号
<i>seqno</i>	<i>integer</i>	项被处理的顺序（1..n）
<i>name</i>	<i>text</i>	配置参数名
<i>setting</i>	<i>text</i>	被赋予给参数的值
<i>applied</i>	<i>boolean</i>	为真表示值已被成功应用
<i>error</i>	<i>text</i>	如果非空，表示一个错误消息，它说明为什么这个项不能被应用

如果配置文件包含语法错误或者非法参数名，服务器将不会尝试从其中应用 任何设置，并且因此所有的*applied*域都为假。在这种情况下，将会有有一个或者多个行的*error*域为非空， 它们说明了为什么出问题。否则，将尽可能应用每个设置。如果一个设置不能 被应用（例如非法值或者该设置不能在服务器开始后改变），会有一个合适的 消息存储在它的*error*域中。一个项的*applied* 域为假的另一种情况是它被后面一个具有相同参数名的项所覆盖，这种情况不 会被认为是一种错误，因此在*error*域中不会有 错误消息。

1.71. ux_group

视图ux_group显式所有角色的名称和未被标记`rolcanlogin`的成员，它是被用做组的角色集合。

表 1.72. ux_group的列

名称	类型	引用	描述
<i>groname</i>	name	ux_authid.rolname	组名
<i>grosysid</i>	oid	ux_authid.oid	组ID
<i>grolist</i>	oid[]	ux_authid.oid	包含此组中角色ID的一个数组

1.72. ux_hba_file_rules

视图ux_hba_file_rules提供客户端认证配置文件ux_hba.conf内容的摘要。该文件中每个非空、非注释行都会在这个视图中出现一行，行中还有标记表示该规则是否被成功地应用。

这个视图可用来检查认证配置文件中按计划的更改是否起作用，或者诊断之前的失败。注意这个视图报告的是该文件的当前内容，而不是服务器最后一次载入的内容。

默认情况下，只有超级用户可以读取ux_hba_file_rules视图。

表 1.73. ux_hba_file_rules的列

名称	列	描述
<i>line_number</i>	<i>integer</i>	这条规则在ux_hba.conf中的行号
<i>type</i>	<i>text</i>	连接类型
<i>database</i>	<i>text[]</i>	这条规则应用的数据库名列表
<i>user_name</i>	<i>text[]</i>	这条规则应用的用户及组名列表
<i>address</i>	<i>text</i>	主机名或IP地址，或者all、samehost、samenet之一，对于本地连接为空
<i>netmask</i>	<i>text</i>	IP地址掩码，如果不适用则为空
<i>auth_method</i>	<i>text</i>	认证方法
<i>options</i>	<i>text[]</i>	为认证方法指定的选项（如果有）
<i>error</i>	<i>text</i>	如果非空，则是一个错误消息，它表示为什么这一行无法被处理

通常，反映一个不正确项的行只有`line_number`和`error`域中有值。

1.73. ux_indexes

视图ux_indexes提供对于数据库中每一个索引信息的访问。

表 1.74. ux_indexes的列

名称	类型	引用	描述
<i>schemaname</i>	name	ux_namespace.nspname	包含表和索引的模式名
<i>tablename</i>	name	ux_class.relname	此索引的基表的名字
<i>indexname</i>	name	ux_class.relname	索引名
<i>tablespace</i>	name	ux_tablespace.spcname	包含索引的表空间名 (如果是数据库的默认值则为空)
<i>indexdef</i>	text		索引定义 (CREATE INDEX命令的重构)

1.74. ux_locks

视图ux_locks提供了数据库服务器上活动进程中保持的锁的信息。

ux_locks中对每一个活动可锁对象、请求锁模式和相关进程的组合都有一行。因此，如果多个进程持有或者正在等待一个可锁对象上的锁，同一个可锁对象可能出现很多次。但是，一个当前没有被锁的对象根本不会出现。

有多种不同类型的可锁对象：整个关系（如表）、关系的单个页、关系的单个元组、事务ID（包括虚拟和永久ID）和普通数据库对象（由类OID和对象OID标识，和ux_description或ux_depend中的相同方式）。扩展一个关系的权力也被表示为一个独立的可锁对象。“咨询”锁可以具有用户定义的意义。

表 1.75. ux_locks的列

名称	类型	引用	描述
<i>locktype</i>	text		可锁对象的类型： relation, extend, page, tuple, transactionid, virtualxid, object, userlock或 advisory
<i>database</i>	oid	ux_database.oid	锁目标存在的数据库的OID，如果目标是一个共享对象则为0，如果目标是一个事务ID则为空
<i>relation</i>	oid	ux_class.oid	作为锁目标的关系的OID，如果目标不是一个关系或者只是关系的一部分则此列为空
<i>page</i>	integer		作为锁目标的页在关系中的页号，如果目标不是一个关系页或元组则此列为空
<i>tuple</i>	smallint		作为锁目标的元组在页中的元组号，如果目标

名称	类型	引用	描述
			不是一个元组则此列为空
<i>virtualxid</i>	text		作为锁目标的事务虚拟ID，如果目标不是一个虚拟事务ID则此列为空
<i>transactionid</i>	xid		作为锁目标的事务ID，如果目标不是一个事务ID则此列为空ID
<i>classid</i>	oid	ux_class.oid	包含锁目标的系统目录的OID，如果目标不是一个普通数据库对象则此列为空
<i>objid</i>	oid	任意OID列	锁目标在它的系统目录中的OID，如果目标不是一个普通数据库对象则为空
<i>objsubid</i>	smallint		锁的目标列号（ <i>classid</i> 和 <i>objid</i> 指表本身），如果目标是某种其他普通数据库对象则此列为0，如果目标不是一个普通数据库对象则此列为空
<i>virtualtransaction</i>	text		保持这个锁或者正在等待这个锁的事务的虚拟ID
<i>pid</i>	integer		保持这个锁或者正在等待这个锁的服务器进程的PID，如果此锁被一个预备事务所持有则此列为空
<i>mode</i>	text		此进程已持有或者希望持有的锁模式
<i>granted</i>	boolean		如果锁已授予则为真，如果锁被等待则为假
<i>fastpath</i>	boolean		如果锁通过快速路径获得则为真，通过主锁表获得则为假

一个行的*granted*为真表示一个被指定进程持有的锁。为假表示该进程当前正在等待获取这个锁，这意味着至少一个其他进程正持有或等待同一个可锁对象上的一个冲突锁。该等待进程将一直休眠直到其他锁被释放（或者一个死锁状态被检测到）。单个进程在同一时间只能等待最多一个锁。

贯穿一个事务的运行，一个服务器进程在其生存周期内都持有一个在其虚拟事务ID上的排他锁。如果一个永久ID被分配给事务（通常发生在事务改变数据库状态时），它也会持有一个在其永久事务ID上的排他锁直到它结束。当一个事务发现它需要等待另一个事务，它也会尝试获取其他事务ID上的共享锁（不管是虚拟还是永久ID，视情况而定）。这只有当其他进程终止并释放其锁后才会成功。

尽管元组是一种可锁对象，关于行级锁的信息被存储在磁盘而不是内存中，因此行级锁通常不在这个视图中出现。如果一个进程正在等待一个行级锁，它通常在这个视图出现，并且表示形式为正在等待已持有该行级锁的永久事务ID上的锁。

咨询锁可以在由一个单一bigint值或两个整形值构成的键上获取。一个bigint键被显示为其高位部分在classid列中，低位部分在objid列中，并且objsubid等于1。原来的bigint值可以使用表达式(classid::bigint << 32) | objid::bigint重组。整形键被显示为第一个键在classid列中，第二个键在objid列中，并且objsubid等于2。键的实际意义由用户决定。咨询锁是每一个数据库的本地锁，所以database列对于一个咨询锁没有意义。

ux_locks提供了一个对于整个数据集簇中所有锁的全局视图，而不仅仅是与当前数据库相关的锁。尽管它的relation列可以被连接到ux_class.oid来标识被锁关系，但这种方法只有在关系属于当前数据库（database列是当前数据库OID或者0的锁对应的关系）的情况下才会得到正确的结果。

pid列可以被连接到 ux_stat_activity视图的pid列来得到持有或等待持有每一个锁的会话的信息。例如

```
SELECT * FROM ux_locks pl LEFT JOIN ux_stat_activity psa
ON pl.pid = psa.pid;
```

另外，如果正在使用预备事务，virtualtransaction列可以被连接到ux_prepared_xacts视图的transaction列来得到持有该锁的预备事务的信息（一个预备事务不可能正在等待一个锁，但它在运行中会一直持有已获得的锁）。例如：

```
SELECT * FROM ux_locks pl LEFT JOIN ux_prepared_xacts ppx
ON pl.virtualtransaction = '-1' || ppx.transaction;
```

虽然通过自连接ux_locks可以获得哪些进程阻塞了其他哪些进程的信息，但是很难得到其中的细节。这样一个查询隐藏了关于哪些锁模式与其他哪些锁模式冲突的知识。更糟糕的是，ux_locks视图无法给出所等待队列中进程的等待顺序，也无法显示哪些进程是代表其他客户端会话运行的并行工作者。更好的方法是使用ux_blocking_pids()函数来标识一个等待进程是被哪些进程阻塞的。

ux_locks视图显示来自于普通锁管理器和谓词锁管理器的数据，它们是独立的系统。此外，普通锁管理器把它的锁分为普通锁和快速路径锁。这些数据并不被保证是完全一致的。当视图被查询时，快速路径锁上的数据（fastpath = true）会被一次性从每一个后端收集起来，且并不冻结整个锁管理器的状态。因此有可能某些锁在上述信息被收集的过程中被获得或者释放。注意，不管怎样这些锁是已知不会和任何当前正在发生的锁冲突。在所有后端已经查询了快速路径锁后，普通锁管理器的剩余部分被作为一个单元锁住，并且所有剩余锁的一个一致快照被作为一个原子动作收集。在解锁普通锁管理器后，谓词锁管理器也被类似地锁住并且所有谓词锁被作为一个原子动作收集。因此，在快速路径锁这种特殊情况下，每一个锁管理器会传递一个一致的结果组。但由于我们并不会同时锁上两个锁管理器，在我们询问完普通锁管理器后或者询问谓词锁管理器之前，锁可以被获得或者释放。

如果对此视图频繁地访问，对普通或者谓词锁管理器加锁可能会对数据库性能产生一定影响。虽然这些锁只会在最少的时间被保持（足以从锁管理器获得数据），但这无法完全消除可能产生的性能影响。

1.75. ux_matviews

视图ux_matviews提供了关于数据库中每一个物化视图的信息。

表 1.76. ux_matviews的列

名称	类型	引用	描述
<i>schemaname</i>	name	ux_namespace.nspname	包含物化视图的模式的名字
<i>matviewname</i>	name	ux_class.relname	物化视图的名字
<i>matviewowner</i>	name	ux_authid.rolname	物化视图拥有者的名字
<i>tablespace</i>	name	ux_tablespace.spcname	包含物化视图的表空间名（如使用数据库默认表空间则为空）
<i>hasindexes</i>	boolean		如果物化视图有（或者最近有过）任何索引，则此列为真
<i>ispopulated</i>	boolean		如果物化视图当前已被填充，则此列为真
<i>definition</i>	text		物化视图的定义（一个重构的SELECT查询）

1.76. ux_policies

视图ux_policies提供了有关数据库中行级 安全性策略的信息。

表 1.77. ux_policies的列

名称	类型	引用	描述
<i>schemaname</i>	name	ux_namespace.nspname	包含策略所在表的模式的名称
<i>tablename</i>	name	ux_class.relname	策略所在表的名称
<i>policyname</i>	name	ux_policy.polname	策略名称
<i>polpermissive</i>	text		策略是宽容性的还是限制性的？
<i>roles</i>	name[]		这个策略适用的角色
<i>cmd</i>	text		这个策略适用的命令类型
<i>qual</i>	text		作为这个策略适用的查询的安全屏障条件增加的表达式
<i>with_check</i>	text		作为尝试向该表增加行的查询的 WITH CHECK 条件增加的表达式

1.77. ux_prepared_statements

ux_prepared_statements视图显示在当前会话中可用的所有预备语句。

`ux_prepared_statements`为每一个预备语句包含一行。当一个新的预备语句被创建时在此视图中会增加一行，反之当一个预备语句被释放时在此视图中会删除一行（例如，通过`DEALLOCATE`命令）。

表 1.78. `ux_prepared_statements`的列

名字	类型	描述
<i>name</i>	text	预备语句的标识符
<i>statement</i>	text	客户端提交用于创建此预备语句的查询语句。对于通过SQL创建的预备语句，这里是由客户端提交的 <code>PREPARE</code> 语句。对于通过前端/后端协议创建的预备语句，这里是预备语句本身的文本。
<i>prepare_time</i>	timestampz	预备语句被创建的时间
<i>parameter_types</i>	regtype[]	预备语句期望的参数类型，以一个regtype数组的形式。这个数组中一个元素所对应的OID可通过将regtype值转换为oid获得。
<i>from_sql</i>	boolean	如果预备语句通过SQL命令 <code>PREPARE</code> 创建，则为true；如果预备语句通过前端/后端协议创建，则为false

`ux_prepared_statements`视图为只读。

1.78. `ux_prepared_xacts`

视图`ux_prepared_xacts`显示关于两阶段提交的当前准备好事务的信息。

`ux_prepared_xacts`为每一个预备事务包含一行。当事务被提交或回滚时，相应的项将被移除。

表 1.79. `ux_prepared_xacts`的列

名称	类型	引用	描述
<i>transaction</i>	xid		预备事务的数字事务标识符
<i>gid</i>	text		分配给事务的全局标识符
<i>prepared</i>	timestamp with time zone		此事务为提交准备好的时间
<i>owner</i>	name	ux_authid .rolname	执行此事务的用户名
<i>database</i>	name	ux_database .datname	执行此事务所在数据库的名字

当`ux_prepared_xacts`视图被访问时，内部事务管理器数据结构被暂时地锁住，并为视图的显示产生一个副本。这确保了视图中是一组一致的结果，并且不会阻塞普通操作。不管怎样，当此视图被频繁访问时，会对数据库性能有所影响。

1.79. ux_publication_tables

视图ux_publication_tables提供publication与其所包含的表之间的映射信息。和底层的目录ux_publication_rel不同，这个视图展开了定义为FOR ALL TABLES的publication，这样对这类publication来说，每一个合格的表都有一行。

表 1.80. ux_publication_tables的列

名称	类型	引用	描述
<i>pubname</i>	name	ux_publication .pubname	publication名称
<i>schemaname</i>	name	ux_namespace .nspname	包含表的方案名称
<i>tablename</i>	name	ux_class .relname	表名

1.80. ux_replication_origin_status

ux_replication_origin_status视图包含有关一个特定源已经重放了多少的信息。

表 1.81. ux_replication_origin_status的列

名称	类型	引用	描述
<i>local_id</i>	oid	ux_replication_origin .roident	内部的节点标识符
<i>external_id</i>	text	ux_replication_origin .roname	外部的节点标识符
<i>remote_lsn</i>	ux_lsn		源节点的 LSN，到这个位置的数据都已经被复制。
<i>local_lsn</i>	ux_lsn		这个节点的 LSN，remote_lsn已经被复制到这里。使用异步提交时，在将数据持久化到磁盘前用它来刷入提交记录。

1.81. ux_replication_slots

ux_replication_slots视图提供了当前存在于数据库集簇上的所有复制槽的列表，其中也包括复制槽的当前状态。

表 1.82. ux_replication_slots的列

名称	类型	引用	描述
<i>slot_name</i>	name		一个唯一的、集簇范围内的复制槽标识符
<i>plugin</i>	name		包含这个逻辑槽正在使用的输出插件的共享对象基础名称，这个列对于物理槽为空值。

名称	类型	引用	描述
<i>slot_type</i>	text		槽类型 - physical （物理）或者 logical （逻辑）
<i>datoid</i>	oid	ux_database.oid	与这个槽相关的数据库的OID，或者为空值。只有逻辑槽具有相关的数据库。
<i>database</i>	text	ux_database.datname	与这个槽相关的数据库的名称，或者为空值。只有逻辑槽具有相关的数据库。
<i>temporary</i>	boolean		如果这是一个临时复制槽则为真。临时槽不会被保存在磁盘上并且会在出错或会话结束时自动被删除掉。
<i>active</i>	boolean		如果这个槽当前正在被使用则为真
<i>active_pid</i>	integer		如果槽当前正在被使用，则记录使用这个槽的会话的进程 ID。如果槽没有被使用则为NULL。
<i>xmin</i>	xid		这个槽需要数据库保留的最旧事务。 VACUUM 不能移除被其后续事务删除的元组。
<i>catalog_xmin</i>	xid		这个槽需要数据库保留的影响系统目录的最旧事务。 VACUUM 不能移除被其后续事务删除的目录元组。
<i>restart_lsn</i>	ux_lsn		可能仍被这个槽的消费者要求的最旧WAL地址（LSN），并且因此不会在检查点期间自动被移除。如果这个槽的LSN从未被保留过，则为NULL。
<i>confirmed_flush_lsn</i>	ux_lsn		代表逻辑槽的消费者已经确认接收数据到什么位置的地址（LSN）。比这个地址更旧的数据已经不再可用。对于物理槽这里是NULL。

1.82. ux_roles

视图ux_roles提供了关于数据库角色的信息。这是ux_authid的一个公共可读视图，它隐去了口令域。

表 1.83. ux_roles的列

名称	类型	引用	描述
<i>rolname</i>	name		角色名
<i>rolsuper</i>	bool		角色是否具有超级用户权限？
<i>rolinherit</i>	bool		如果此角色是另一个角色的成员，角色是否能自动继承另一个角色的权限？
<i>rolcreatorole</i>	bool		角色能否创建更多角色？
<i>rolcreatedb</i>	bool		角色能否创建数据库？
<i>rolcanlogin</i>	bool		角色是否能登录？即此角色能否被作为初始会话授权标识符？
<i>rolreplication</i>	bool		角色是一个复制角色。复制角色可以开启复制连接并且创建和删除复制槽。
<i>rolconnlimit</i>	int4		对于一个可登录的角色，这里设置角色可以发起的最大并发连接数。-1表示无限制。
<i>rolpassword</i>	text		不是口令（看起来是*****）
<i>rolvaliduntil</i>	timestampz		口令失效时间（只用于口令认证），如果永不失效则为空
<i>rolbypassrls</i>	bool		绕过每一条行级安全性策略的角色。
<i>rolconfig</i>	text[]		运行时配置变量的角色特定默认值
<i>oid</i>	oid	ux_authid.oid	角色的ID

1.83. ux_rules

视图ux_rules提供对查询重写规则的信息访问。

表 1.84. ux_rules的列

名称	类型	引用	描述
<i>schemaname</i>	name	ux_namespace.nspname	包含表的模式名
<i>tablename</i>	name	ux_class.relname	规则适用的表名
<i>rulename</i>	name	ux_rewrite.rulename	规则名
<i>definition</i>	text		规则定义（创建命令的重构）

ux_rules视图排除了视图和物化视图的ON SELECT规则，它们可以在ux_views和ux_matviews中找到。

1.84. ux_seclabels

视图ux_seclabels提供对安全标签的信息访问。它是[ux_seclabel](#)目录的一个便于查询的版本。

表 1.85. ux_seclabels的列

名称	类型	引用	描述
<i>objoid</i>	oid	任意OID列	安全标签所属对象的OID
<i>classoid</i>	oid	ux_class.oid	对象出现的系统目录的OID
<i>objsubid</i>	int4		对于一个表列上的安全标签，这里是列号（ <i>objoid</i> 和 <i>classoid</i> 指表本身）。对于所有其他对象类型，此列为0。
<i>objtype</i>	text		此标签应用的对象类型，以文本方式。
<i>objnamespace</i>	oid	ux_namespace.oid	如果适用，为此对象的名字空间的OID；否则为空。
<i>objname</i>	text		此标签应用的对象名，以文本形式。
<i>provider</i>	text	ux_seclabel.provider	与此标签相关的标签提供者。
<i>label</i>	text	ux_seclabel.label	应用于此对象的安全标签。

1.85. ux_sequences

视图ux_sequences提供对数据库中每个序列的信息的访问。

表 1.86. ux_sequences的列

名称	类型	引用	描述
<i>schemaname</i>	name	ux_namespace.nspname	包含序列的方案名

名称	类型	引用	描述
<i>sequencename</i>	name	ux_class.relname	序列的名称
<i>sequenceowner</i>	name	ux_authid.rolname	序列的拥有者的名称
<i>data_type</i>	regtype	ux_type.oid	序列的数据类型
<i>start_value</i>	bigint		序列的起始值
<i>min_value</i>	bigint		序列的最小值
<i>max_value</i>	bigint		序列的最大值
<i>increment_by</i>	bigint		序列的增量值
<i>cycle</i>	boolean		序列是否循环
<i>cache_size</i>	bigint		序列的缓冲尺寸
<i>last_value</i>	bigint		最后一个被写入到磁盘的序列值。如果使用了缓冲，这个值可能比从序列中取出的最后一个值大。如果还没有从该序列读取过，则为空。此外，如果当前用户没有该序列上的USAGE或SELECT特权，则这个值为空。

1.86. ux_settings

视图ux_settings提供了对服务器上运行时参数的访问。它本质上是SHOW和SET命令的可替换接口。它还提供了SHOW不能提供的关于每一个参数的一些现实，例如最大值和最小值。

表 1.87. ux_settings的列

名字	类型	描述
<i>name</i>	text	运行时配置参数名
<i>setting</i>	text	参数的当前值
<i>unit</i>	text	参数的隐式单元
<i>category</i>	text	参数的逻辑组
<i>short_desc</i>	text	参数的简短描述
<i>extra_desc</i>	text	附加的参数的详细描述
<i>context</i>	text	要求设置此参数值的上下文
<i>vartype</i>	text	参数类型（bool、enum、integer、real或string）
<i>source</i>	text	当前参数值的来源
<i>min_val</i>	text	参数的最小允许值（对非数字值为空）
<i>max_val</i>	text	参数的最大允许值（对非数字值为空）

名字	类型	描述
<i>enumvals</i>	text[]	一个枚举参数的允许值（对非数字值为空）
<i>boot_val</i>	text	如果参数没有被别的其他设置，此列为在服务器启动时设定的参数值
<i>reset_val</i>	text	在当前会话中， RESET 将会设置的参数值
<i>sourcefile</i>	text	当前值被设置的配置文件（空值表示从非配置文件的其他来源设置，由不是超级用户也不是 ux_read_all_settings 成员的用户检查时也为空值），在配置文件中 include 指令时有用
<i>sourceline</i>	integer	当前值被设置的配置文件中的行号（空值表示从非配置文件的其他来源设置，由不是超级用户也不是 ux_read_all_settings 成员的用户检查时也为空值）。
<i>pending_restart</i>	boolean	如果配置文件中修改了该值但需要重启，则为 true ，否则为 false 。

对于*context*有多种可能的取值。为了降低改变设置的难度，它们是：

internal

这些设置不能被直接修改，它们反映了内部决定的值。某些可能在使用不同配置选项重建系统时或者改变**initdb**的选项时可以调整。

postmaster

这些设置只能在服务器启动时应用，因此任何修改都需要重启服务器。这些设置的值通常都存储在**uxsino.conf**文件中，或者在启动服务器时通过命令行传递。当然，具有更低*context*类型的设置也可以在服务器启动时间被设置。

sighup

对于这些设置的修改可以在**uxsino.conf**中完成并且不需要重启服务器。发送一个**SIGHUP**信号给**postmaster**会导致它重新读取**uxsino.conf**并应用修改。**Postmaster**将会把**SIGHUP**信号传递给它的孩子进程，这样它们也会获得新的值。

superuser-backend

对于这些设置的更改可以在**uxsino.conf**中进行而无需重启服务器。也可以在连接请求包（例如通过**libpq**的**PGOPTIONS**环境变量）中为一个特定的会话设定它们，但是只有在连接用户是超级用户时才能这样做。如果，在会话启动后这些设置就不会改变。如果你在**uxsino.conf**改变了它们，向**postmaster**发送一个**SIGHUP**信号让**postmaster**重新读取**uxsino.conf**。新的值将只会影响后续启动的会话。

backend

对于这些设置的修改可以在`uxsino.conf`中完成并且不需要重启服务器。它们也可以在一个连接请求包（例如，通过libpq的PGOPTIONS环境变量）中为一个特定会话设置，任何用户都可以为这个会话做这种修改。然而，这些设置在会话启动后永不变化。如果你在`uxsino.conf`中修改它们，可以向postmaster发送一个SIGHUP信号让它重读`uxsino.conf`。新值只会影响后续启动的会话。

superuser

这些设置可以从`uxsino.conf`设置，或者在会话中用SET命令设置。仅当没有通过SET设置会话本地值时，`uxsino.conf`中的改变才会影响现有的会话。

user

这些设置可以从`uxsino.conf`设置，或者在会话中用SET命令设置。任何用户都被允许修改它们的会话本地值。仅当没有通过SET设置会话本地值时，`uxsino.conf`中的改变才会影响现有的会话。

`ux_settings`视图不能被插入或者从中删除，但是它可以被更新。在`ux_settings`的一行上的一个UPDATE等价于在该参数上执行一个SET命令。修改将只会影响当前会话使用的值。如果一个UPDATE在一个后来中断的事务中被发出，UPDATE命令的效果也会随着事务的回滚而消失。一旦所在的事务被提交，效果将一直保持到会话结束，除非有其他UPDATE或SET重新修改它。

1.87. ux_shadow

视图`ux_shadow`显示`ux_authid`中所有被标记为`rolcanlogin`的角色的属性。

由于这个表包含口令，所以不能是公众可读的，这也是采用`ux_shadow`这个名字的原因。而`ux_user`是`ux_shadow`上的一个公共可读视图，它屏蔽了口令域。

表 1.88. `ux_shadow`的列

名称	类型	引用	描述
<i>username</i>	name	ux_authid.rolname	用户名
<i>usesysid</i>	oid	ux_authid.oid	用户的ID
<i>usecreatedb</i>	bool		用户能否创建数据库
<i>usesuper</i>	bool		用户是否为一个超级用户
<i>userepl</i>	bool		用户能否开启流复制并将系统设置或者取消备份模式。
<i>usebypassrls</i>	bool		用户能否绕过所有的行级安全性策略。
<i>passwd</i>	text		口令（可能被加密），如果没有则为空。关于加密口令如何存储请参见 ux_authid 。
<i>valuntil</i>	timestampz		口令过期时间（仅用于口令认证）

名称	类型	引用	描述
<i>useconfig</i>	text[]		运行时配置变量的会话默认值

1.88. ux_stats

视图ux_stats提供对存储在[ux_statistic](#)目录中信息的访问。此视图能访问ux_statistic行是有限制的，可访问行所对应的表必须是用户有读权限的。因此让所有用户都可以读此视图是安全的。

ux_stats也被设计为能以更适合阅读的格式显示底层目录的信息—但代价是只要为ux_statistic定义了新的槽类型，就必须扩展此视图的模式。

表 1.89. ux_stats的列

名称	类型	引用	描述
<i>schemaname</i>	name	ux_namespace.nspname	包含表的模式名
<i>tablename</i>	name	ux_class.relname	表名
<i>attname</i>	name	ux_attribute.attname	被此行描述的列名
<i>inherited</i>	bool		如果为真，表示此行包括继承子列，不仅仅是指定表中的值
<i>null_frac</i>	real		列项中为空的比例
<i>avg_width</i>	integer		列项的平均字节宽度
<i>n_distinct</i>	real		如果大于零，表示列中可区分值的估计个数。如果小于零，是可区分值个数除以行数的负值（当ANALYZE认为可区分值的数量会随着表增长而增加时采用负值的形式，而如果认为列具有固定数量的可选值时采用正值的形式）。例如，-1表示一个唯一列，即其中可区分值的个数等于行数。
<i>most_common_vals</i>	anyarray		列中最常用值的一个列表（如果没有任何一个值看起来比其他值更常用，此列为空）
<i>most_common_freqs</i>	real[]		最常用值的频率列表，即每一个常用值的出现次数除以总行数（如果 <i>most_common_vals</i> 为空，则此列为空）
<i>histogram_bounds</i>	anyarray		将列值划分成大小接近的组的值列表。如果存

名称	类型	引用	描述
			在 <code>most_common_vals</code> ，其中的值会被直方图计算所忽略（如果列类型没有一个<操作符或者 <code>most_common_vals</code> 等于整个值集合，则此列为空）
<code>correlation</code>	real		物理行顺序和列值逻辑顺序之间的统计关联。其范围从-1到+1。当值接近-1或+1时，在列上的一个索引扫描被认为比值接近0时的代价更低，因为这种情况减少了对磁盘的随机访问（如果列数据类型不具有一个<操作符，则此列为空）
<code>most_common_elems</code>	anyarray		在列值中，最经常出现的非空元素列表（对标度类型为空）
<code>most_common_elem_freqs</code>	real[]		最常用元素值的频度列表，即含有至少一个给定值实例的行的分数。在每个元素的频度之后有二至三个附加值，它们是每个元素频度的最小和最大值，以及可选的空元素的频度（如果 <code>most_common_elems</code> 为空，则此列为空）
<code>elem_count_histogram</code>	real[]		在列值中可区分非空元素值计数的一个直方图，后面跟随可区分非空元素的平均数（对于标度类型为空）

在数组域中项的最大数目可以使用`ALTER TABLE SET STATISTICS`命令控制，或者设置`default_statistics_target`运行时参数从全局上进行控制

1.89. ux_stats_ext

视图`ux_stats_ext`提供了访问存储在`ux_statistic_ext`和`ux_statistic_ext_data`目录中的信息的手段。该视图仅允许访问`ux_statistic_ext`和`ux_statistic_ext_data`的行，这些行对应于用户有权读取的表，因此允许公众读取该视图是安全的。

`ux_stats_ext`也设计成以比底层目录更可读的格式来展示信息—代价是每当有扩展的统计信息的新类型加到`ux_statistic_ext`中时其模式必须扩展。

表 1.90. ux_stats_ext的列

名称	类型	引用	描述
<i>schemaname</i>	name	ux_namespace.nspname	包含表的模式名
<i>tablename</i>	name	ux_class.relname	表名
<i>statistics_schemaname</i>	name	ux_namespace.nspname	包含扩展的统计信息的模式名
<i>statistics_name</i>	name	ux_statistic_ext.stxname	扩展的统计信息的名称
<i>statistics_owner</i>	oid	ux_authid.oid	扩展的统计信息对的拥有者
<i>attnames</i>	name[]	ux_attribute.attname	扩展的统计信息定义所在的列名称
<i>kinds</i>	text[]		为此记录启用的扩展统计信息类型
<i>n_distinct</i>	ux_ndistinct		列值组合的N-不同计数。如果大于零，则为组合中不同值的估计数量。如果小于零，则为不同值数量的负数除以行数。（当ANALYZE认为随着表的增长不同值的数量可能会增加时，使用负数形式；当该列似乎具有固定数量的可能值时，则使用正数形式。）例如，-1表示列的唯一组合，其中不同组合的数量与行数相同。
<i>dependencies</i>	ux_dependencies		功能的依赖关系统计信息
<i>most_common_vals</i>	anyarray		列中值的最常见组合的列表（如果没有组合看上去比其它的更常见，则为空。）
<i>most_common_val_nulls</i>	anyarray		值最常见组合的NULL标志的列表 （当 <i>most_common_vals</i> 是空值时，为空。）
<i>most_common_freqs</i>	real[]		最常见组合的频率的列表，即每个出现的数量除以行的总数 （当 <i>most_common_vals</i> 是空值时，为空）
<i>most_common_base_freqs</i>	real[]		最常见组合的基本频率的列表，即每个值频率的乘积。

名称	类型	引用	描述
			(当 <code>most_common_vals</code> 是空值时, 为空。)

数组字段条目的最大数量可以基于一列列地使用`ALTER TABLE SET STATISTICS`命令来控制, 或者通过设置全局的`default_statistics_target`运行时参数来控制。

1.90. ux_tables

视图`ux_tables`提供对数据库中每个表的信息的访问。

表 1.91. `ux_tables`的列

名称	类型	引用	描述
<i>schemaname</i>	name	ux_namespace.nspname	包含表的模式名
<i>tablename</i>	name	ux_class.relname	表名
<i>tableowner</i>	name	ux_authid.rolname	表拥有者的名字
<i>tablespace</i>	name	ux_tablespace.spcname	包含表的表空间的名字 (如果使用数据库的默认表空间, 此列为空)
<i>hasindexes</i>	boolean	ux_class.relhasindex	如果表有(或最近有过)任何索引, 此列为真
<i>hasrules</i>	boolean	ux_class.relhasrules	如果表有(或曾经有过)规则, 此列为真
<i>hastriggers</i>	boolean	ux_class.relhastriggers	如果表有(或者曾经有过)触发器, 此列为真
<i>rowsecurity</i>	boolean	ux_class.relrowsecurity	如果表上启用了行安全性则为真

1.91. ux_timezone_abbrevs

视图`ux_timezone_abbrevs`提供了对当前被时间输入例程识别的时区缩写的列表。当`timezone_abbreviations`运行时参数被修改, 此视图的内容会发生变化。

表 1.92. `ux_timezone_abbrevs`的列

名字	类型	描述
<i>abbrev</i>	text	时区缩写
<i>utc_offset</i>	interval	相对于UTC的偏移(正值表示格林威治东部)
<i>is_dst</i>	boolean	如果这是一个夏令时缩写, 则为真

虽然大部分时区缩写表示从 UTC 开始的固定偏移, 但是有一些在历史上有值的变化。在这种情况下, 这个视图表示它们现在的含义。

1.92. ux_timezone_names

视图`ux_timezone_names`提供了一个被`SET TIMEZONE`识别的时区名字的列表，以及它们的相关缩写、UTC偏移和夏令时状态（从技术上来说，UXsinoDB不使用UTC是因为闰秒没有被处理）。和[ux_timezone_abbrevs](#)中展示的缩写不同，这里很多名字隐含了一组夏令时转换日期规则。因此，相关信息在本地DST边界间变化。所显示的信息基于`CURRENT_TIMESTAMP`的当前值计算得来。

表 1.93. `ux_timezone_names`的列

名字	类型	描述
<i>name</i>	text	时区名
<i>abbrev</i>	text	时区缩写
<i>utc_offset</i>	interval	相对于UTC的偏移（正值表示格林威治东部）
<i>is_dst</i>	boolean	如果当前保持为夏令时则为真

1.93. `ux_user`

视图`ux_user`提供关于数据库用户的信息。这是[ux_shadow](#)的一个公共可读的视图，它消除了口令域。

表 1.94. `ux_user`的列

名字	类型	描述
<i>username</i>	name	用户名
<i>usesysid</i>	oid	用户的ID
<i>usecreatedb</i>	bool	用户是否能创建数据库
<i>usesuper</i>	bool	用户是否为超级用户
<i>userepl</i>	bool	用户能否开启流复制以及将系统转入/转出备份模式。
<i>usebypassrls</i>	bool	用户能否绕过所有的行级安全性策略。
<i>passwd</i>	text	不是口令（总是显示为*****）
<i>valuntil</i>	timestampzt	口令过期时间（只用于口令认证）
<i>useconfig</i>	text[]	运行时配置变量的会话默认值

1.94. `ux_user_mappings`

视图`ux_user_mappings`提供有关用户映射的信息。这是[ux_user_mapping](#)的一个公共可读视图，它对无权使用的用户省去了选项域。

表 1.95. `ux_user_mappings`的列

名称	类型	引用	描述
<i>umid</i>	oid	ux_user_mapping.oid	用户映射的OID

名称	类型	引用	描述
<i>srvid</i>	oid	ux_foreign_server.oid	包含该映射的外部服务器的OID
<i>srvname</i>	name	ux_foreign_server.srvname	外部服务器名
<i>umuser</i>	oid	ux_authid.oid	将被映射的本地角色的OID，如果用户映射是公共的则为0
<i>username</i>	name		将被映射的本地用户名
<i>umoptions</i>	text[]		用户映射指定选项，以“keyword=value”字符串的形式

为了保护存储为用户映射选项的口令信息，*umoptions*列将被读作空，除非满足下列情况之一：

- 当前用户就是被映射的用户，并且拥有该服务器或者持有其上的USAGE特权
- 当前用户是服务器的所有者并且映射是用于PUBLIC
- 当前用户是一个超级用户

1.95. ux_views

视图ux_views提供了数据库中每个视图的信息。

表 1.96. ux_views的列

名称	类型	引用	描述
<i>schemaname</i>	name	ux_namespace.nspname	包含视图的模式名
<i>viewname</i>	name	ux_class.relname	视图名称
<i>viewowner</i>	name	ux_authid.rolname	视图拥有者的名字
<i>definition</i>	text		视图定义（一个重构的SELECT查询）

1.96. 兼容oracle视图

表 1.97. 兼容oracle视图

视图名字	用途
all_arguments	当前用户可以访问的函数和过程的参数
all_col_comments	当前用户可访问的所有序列
all_col_privs	当前用户下可以查看的所有列级权限
all_cons_columns	当前用户可访问的所有约束的列
all_constraints	当前用户可以访问的表上的约束定义
all_ind_columns	当前用户所能获取到的表上建有索引的列信息
all_indexes	当前用户所能获取到的表上的索引信息
all_objects	当前用户下可以查看的所有对象
all_sequences	当前用户可访问的所有序列

视图名字	用途
all_source	当前用户可访问的存储对象的文本源
all_synonyms	当前用户所能查看的同义词信息。包括： <ul style="list-style-type: none"> • 当前用户的私有同义词 • 所有的public同义词 • 本地同义词（DB_LINK列为空）所指向，或者嵌套指向的对象，当前用户有权限查看
all_tab_cols	当前用户可以访问的表，视图的列
all_tab_columns	当前用户可以访问的表，视图的非隐藏列
all_tab_comments	在当前用户可以访问的表和视图上显示注释
all_tab_privs	描述当前用户，该用户可以是对象的所有者、授予者或者被授予者
all_tables	当前用户可访问的表
all_trigger_cols	当前用户可访问的所有触发器的列
all_triggers	当前用户可访问的所有触发器
all_users	列出当前用户可见的数据库的所有用户
all_views	当前用户所能查看的所有的视图信息
db_files	数据库中所有表空间、表空间使用的存储容量及表空间所在存储设备的容量信息
dba_arguments	列出数据库中可用的过程和函数的参数
dba_col_comments	在数据库中所有表和视图的列上显示注释
dba_col_privs	所有列级权限
dba_cons_columns	约束中指定的数据库中的所有列
dba_constraints	当前用户所拥有的表的所有约束定义
dba_ind_columns	数据库中所有建有索引的列的信息
dba_indexes	数据库中所有索引的信息
dba_objects	当前用户下可以查看的所有对象
dba_role_privs	授予所有用户的角色以及数据库中的角色
dba_roles	列出数据库中存在的角色
dba_sequences	数据库中所有的序列
dba_source	描述数据库中所有存储对象的文本源
dba_synonyms	数据库中所有同义词的信息
dba_tab_cols	当前所在数据库中，所有的表、视图的列信息
dba_tab_columns	当前所在数据库中，所有的表、视图的非隐藏列信息
dba_tab_comments	显示数据库中所有表和视图的注释
dba_tab_privs	描述数据库中的所有对象授予
dba_tables	数据库中的所有表
dba_tablespace	数据库中的表空间

视图名字	用途
dba_tablespaces	数据库中的所有表空间
dba_trigger_cols	数据库中所有触发器的列
dba_triggers	数据库中所有的触发器
dba_users	数据库中所有用户的信息
dba_views	当前用户的所有视图的信息
user_arguments	描述当前用户拥有的过程和函数的参数
user_col_comments	在当前用户拥有的表和视图的列上注释
user_col_privs	当前用户下的列级权限表
user_cons_columns	描述当前用户拥有并在约束中指定的列
user_constraints	当前用户所拥有的表的所有约束定义
user_ind_columns	数据库中所有建有索引的列的信息
user_indexes	当前用户拥有的所有索引信息
user_objects	当前用户拥有的所有对象
user_role_privs	授予当前用户的角色
user_sequences	当前用户的所有序列的信息
user_source	当前用户的所有程序源的信息
user_synonyms	数据库中用户所有同义词的信息
user_tab_cols	当前用户可以访问的表，视图的列
user_tab_columns	数据库中所有表列的信息
user_tab_comments	当前用户拥有的表和视图上的注释
user_tab_privs	当前用户的对象权限，该用户可以是对象的所有者、授予者或者被授予者
user_tables	当前用户的所有表的信息
user_tablespace	描述当前用户可访问的表空间
user_tablespaces	当前用户可访问的表空间
user_trigger_cols	当前用户用的所有触发器的列信息
user_triggers	当前用户的所有触发器的信息
user_users	当前用户
user_views	当前用户的所有视图的信息
V\$DATABASE	当前数据库信息
V\$INSTANCE	当前实例信息
V\$LOCK	列出数据库当前持有的锁以及对锁的未完成请求
V\$LOCKED_OBJECT	当前被锁定的对象
V\$PARAMETER	列出影响当前会话的初始化参数信息
V\$SESSION	当前会话信息
V\$SYSSTAT	当前系统统计信息

1.97. all_arguments

all_arguments视图描述当前用户可以访问的函数和过程的参数。

表 1.98. all_arguments的列

名称	类型	描述
OWNER	CHARACTER VARYING	对象的拥有者
PACKAGE_NAME	CHARACTER VARYING	包的名称
OBJECT_NAME	CHARACTER VARYING	过程或函数的名称
OBJECT_ID	OID	对象的对象号
POSITION	INTEGER	此列保存该项在参数列表中的位置，0表示函数返回值
ARGUMENT_NAME	CHARACTER VARYING	参数名称，空参数名用于表示函数返回
IN_OUT	CHARACTER VARYING	入参/出参类型 <ul style="list-style-type: none"> • IN • OUT • IN/OUT
TYPE_NAME	CHARACTER VARYING	实参类型的名称。如果类型是包本地类型(也就是说，它在包规范中声明)，则此列显示包的名称
OVERLOAD	CHARACTER VARYING	指示第n个重载(按其在源中的出现顺序)；否则，它是NULL
SUBPROGRAM_ID	INTEGER	唯一子程序标识符
SEQUENCE	INTEGER	定义参数的顺序。参数序列从1开始。首先是返回类型，然后是每个参数
DATA_LEVEL	INTEGER	复合类型实参的嵌套深度
DATA_TYPE	CHARACTER VARYING	参数的数据类型
DEFAULTED	CHARACTER VARYING	指定参数是否为默认值
DEFAULT_VALUE	TEXT	保留供将来使用
DEFAULT_LENGTH	INTEGER	保留供将来使用
DATA_LENGTH	INTEGER	列长度(以字节为单位)
DATA_PRECISION	INTEGER	十进制数字(NUMBER)或二进制数字(FLOAT)的长度
DATA_SCALE	INTEGER	小数点右边的数字
RADIX	INTEGER	一个数字的基数参数
CHARACTER_SET_NAME	CHARACTER VARYING	参数的字符集名称
TYPE_OWNER	CHARACTER VARYING	参数类型的所有者

名称	类型	描述
TYPE_SUBNAME	CHARACTER VARYING	只与包本地类型相关。显示在TYPE_NAME列标识的包中声明的类型的名称
TYPE_LINK	CHARACTER VARYING	当TYPE_NAME列中标识的包是远程包时，仅与包本地类型相关。此列显示用于引用远程包的数据库链接
TYPE_OBJECT_TYPE	CHARACTER VARYING	显示由TYPE_OWNER、TYPE_NAME和TYPE_SUBNAME列描述的类型。取值如下所示。 <ul style="list-style-type: none"> • TABLE • VIEW • PACKAGE • TYPE
PLS_TYPE	CHARACTER VARYING	对于数值形参，实参的PL/SQL类型的名称。否则无效
CHAR_LENGTH	INTEGER	字符串数据类型的字符限制
CHAR_USED	CHARACTER VARYING	指示字符串的字节限制(B)或字符限制(C)是否正式
ORIGIN_CON_ID	INTEGER	数据产生的容器的ID。可能的值包括： <ul style="list-style-type: none"> • 0：该值用于非cdb中的行。此值不用于cdb • n：该值用于包含起源于容器ID为n的容器的数据的行(如果该行起源于root，则n = 1)

1.98. all_col_comments

all_col_comments视图描述当前用户可访问的所有序列。

表 1.99. all_col_comments的列

名称	类型	描述
OWNER	VARCHAR2(63)	对象的拥有者
TABLE_NAME	VARCHAR2(63)	对象名
COLUMN_NAME	VARCHAR2(63)	列名
COMMENTS	VARCHAR2(4000)	描述

1.99. all_col_privs

all_col_privs视图描述当前用户下可以查看的所有列级权限

表 1.100. all_col_privs的列

名称	类型	描述
GRANTOR	VARCHAR2 (63)	执行授权的用户的名称
OWNER	VARCHAR2 (63)	对象的拥有者
GRANTEE	VARCHAR2 (63)	授予访问权的用户或角色的名称
TABLE_CATALOG	VARCHAR2 (63)	当前的数据库
TABLE_SCHEMA	VARCHAR2 (63)	对象的模式
TABLE_NAME	VARCHAR2 (63)	表名
COLUMN_NAME	VARCHAR2 (63)	列名
PRIVILEGE_TYPE	VARCHAR2 (40)	权限名称
IS_GRANTABLE	VARCHAR2 (3)	允许访问

1.100. all_cons_columns

all_cons_columns视图描述当前用户可访问的所有约束的列。

表 1.101. all_cons_columns的列

名称	类型	描述
OWNER	VARCHAR2 (63)	兑现的拥有者
CONSTRAINT_NAME	VARCHAR2 (63)	约束名
TABLE_NAME	VARCHAR2 (63)	表名
COLUMN_NAME	VARCHAR2 (4000)	列名
POSITION	NUMERIC	定义中列的原始位置

1.101. all_constraints

all_constraints视图描述当前用户可以访问的表上的约束定义。

表 1.102. all_constraints的列

名称	类型	描述
OWNER	VARCHAR2 (63)	对象的拥有者
CONSTRAINT_NAME	VARCHAR2 (63)	约束名
CONSTRAINT_TYPE	VARCHAR2 (1)	约束定义的类型，取值如下所示 <ul style="list-style-type: none">• C -表上的检查约束• P -主键

名称	类型	描述
		<ul style="list-style-type: none"> • U -唯一密钥 • R -外键 • T -受限触发器 • X -排他性约束
TABLE_NAME	VARCHAR2 (63)	表名
SEARCH_CONDITION	text	检查约束的搜索条件的文本。只有当行起源于当前容器时，此列才返回正确的值
R_OWNER	VARCHAR2 (63)	引用约束中引用的表的所有者
R_CONSTRAINT_NAME	VARCHAR2 (63)	引用表的唯一约束定义的名称
DELETE_RULE	VARCHAR2 (9)	<p>引用约束的删除规则如下所示。</p> <ul style="list-style-type: none"> • RESTRICT • CASCADE • NULL • SET NULL • SET DEFAULT • NO ACTION
STATUS	VARCHAR2 (8)	<p>约束的实施状态如下所示。</p> <ul style="list-style-type: none"> • ENABLED • DISABLED
DEFERRABLE	VARCHAR2 (14)	指示约束是否可延迟 (deferrable)或不可延迟 (not deferrable)
DEFERRED	VARCHAR2 (9)	指示约束最初是否被延迟 (deferred) (IMMEDIATE)
VALIDATED	VARCHAR2 (13)	<p>当STATUS = ENABLED时，可能的值为：</p> <ul style="list-style-type: none"> • VALIDATED -所有数据都遵守约束 (也就是说，当启用约束时，表中的现有数据以及输入到表中的任何后续数据都将得到验证) • NOVALIDATED -所有数据都可能不遵守约束 (也就是说，当启用约束时，表中现有的数据没有被验证，但是输入到表中的后续数据被验证)

名称	类型	描述
GENERATED	VARCHAR2 (14)	指示约束的名称是用户生成 (USER name) 还是系统生成 (GENERATED name)
BAD	VARCHAR2 (3)	指示此约束是否以模糊的方式指定世纪 (BAD) 或否 (NULL)。为了避免这种模糊性导致的错误，可以使用带四位数年份的 TO_DATE 函数重写约束
RELY	VARCHAR2 (4)	当 VALIDATED = NOT VALIDATED 时，此列表示查询重写 (RELY) 时是否考虑约束 (NULL)
LAST_CHANGE	TIMESTAMP (0) WITHOUT TIME ZONE	约束最后一次启用或禁用的时间
INDEX_OWNER	VARCHAR2 (63)	拥有索引的用户名
INDEX_NAME	VARCHAR2 (63)	索引的名称 (仅在唯一和主键约束时显示)
INVALID	VARCHAR2 (7)	指示约束是否无效 (invalid) 或无效 (NULL)
VIEW_RELATED	VARCHAR2 (14)	指示约束是否依赖于视图 (DEPEND on view) (NULL)

1.102. all_ind_columns

all_ind_columns 视图描述当前用户所能获取到的表上建有索引的列信息。

表 1.103. all_ind_columns 的列

名称	类型	描述
INDEX_OWNER	VARCHAR2 (63)	对象的拥有者
INDEX_NAME	VARCHAR2 (63)	索引名
TABLE_OWNER	VARCHAR2 (63)	表的拥有者
TABLE_NAME	VARCHAR2 (63)	表名
COLUMN_NAME	VARCHAR2 (4000)	列名
COLUMN_POSITION	SMALLINT	索引中的列或属性的位置
COLUMN_LENGTH	NUMERIC	列的索引长度
CHAR_LENGTH	NUMERIC	列的最大码点长度
DESCEND	VARCHAR2 (4)	列是否按降序排序 (Y/N)

1.103. all_indexes

all_indexes 视图描述当前用户所能获取到的表上的索引信息。

表 1.104. all_indexes的列

名称	类型	描述
OWNER	VARCHAR2 (63)	索引的拥有者
INDEX_NAME	VARCHAR2 (63)	索引名
INDEX_TYPE	VARCHAR2 (27)	索引类型
TABLE_OWNER	VARCHAR2 (63)	表的拥有者
TABLE_NAME	VARCHAR2 (63)	表名
TABLE_TYPE	TEXT	表类型
UNIQUENESS	VARCHAR2 (9)	索引的惟一状态
COMPRESSION	VARCHAR2 (8)	用于索引的压缩类型
PREFIX_LENGTH	NUMERIC	压缩键前缀中的列数
TABLESPACE_NAME	VARCHAR2 (63)	包含索引的表空间的名称
INI_TRANS	VARCHAR2 (7)	初始交易数量
MAX_TRANS	VARCHAR2 (7)	最大交易数量
INITIAL_EXTENT	VARCHAR2 (7)	初始范围的大小
NEXT_EXTENT	VARCHAR2 (7)	辅助扩展区的大小
MIN_EXTENTS	VARCHAR2 (7)	段中允许的最小范围数
MAX_EXTENTS	VARCHAR2 (7)	段中允许的最大范围数
PCT_INCREASE	VARCHAR2 (7)	区大小增加的百分比
PCT_THRESHOLD	VARCHAR2 (7)	每个索引条目允许的块空间百分比阈值
INCLUDE_COLUMN	VARCHAR2 (7)	要包含在按索引组织的表主键(非溢出)索引中的最后一列的列ID
FREELISTS	VARCHAR2 (7)	分配给该段的进程空闲列表数
FREELIST_GROUPS	VARCHAR2 (7)	分配给该段的空闲列表组的数量
PCT_FREE	VARCHAR2 (7)	块中可用空间的最小百分比
LOGGING	VARCHAR2 (7)	指示是否记录对索引的更改:
BLEVEL	VARCHAR2 (7)	b *-树级别(从其根块到其叶块的索引深度)。深度0表示根块和叶块是相同的
LEAF_BLOCKS	VARCHAR2 (7)	索引中的叶块数
DISTINCT_KEYS	VARCHAR2 (7)	不同索引值的数量。对于强制执行的索引UNIQUE和PRIMARY KEY约束, 该值与表中的行数相同(*_TABLES.NUM_ROWS)
AVG_LEAF_BLOCKS_PER_KEY	VARCHAR2 (7)	索引中每个不同值出现的叶块数的平均值, 四舍五入为最接近的整数。对于强制执行的索引UNIQUE和PRIMARY KEY约束, 该值始终为1

名称	类型	描述
AVG_DATA_BLOCKS_PER_KEY	VARCHAR2 (7)	表中由索引中的非重复值指向的平均数据块数，四舍五入为最接近的整数。此统计信息是包含索引列中包含给定值的行的数据块的平均数量
CLUSTERING_FACTOR	VARCHAR2 (7)	根据索引值指示表中行的顺序 如果该值接近块数，则该表非常有序。在这种情况下，单个叶块中的索引条目倾向于指向相同数据块中的行 如果该值接近行数，则该表是随机排序的。在这种情况下，同一叶块中的索引条目不太可能指向同一数据块中的行
STATUS	VARCHAR2 (8)	指示非分区索引是否为VALID或者UNUSABLE
NUM_ROWS	VARCHAR2 (7)	索引中的行数。
SAMPLE_SIZE	VARCHAR2 (7)	用于分析指数的样本大小
LAST_ANALYZED	VARCHAR2 (20)	最近分析此索引的日期
DEGREE	VARCHAR2 (40)	每个实例用于扫描索引的线程数，或者DEFAULT
INSTANCES	VARCHAR2 (40)	要扫描索引的实例数，或者DEFAULT
PARTITIONED	VARCHAR2 (3)	指示索引是否已分区 (YES) 还是不 (NO)
TEMPORARY	VARCHAR2 (1)	指示索引是否在临时表上 (Y) 还是不 (N)
GENERATED	VARCHAR2 (1)	指示索引的名称是否是系统生成的 (Y) 还是不 (N)
SECONDARY	VARCHAR2 (1)	指示该索引是否是由 ODCIIndexCreateOracle数据盒式磁带的方法 (Y) 还是不 (N)
BUFFER_POOL	VARCHAR2 (7)	用于索引块的缓冲池：
USER_STATS	VARCHAR2 (3)	指示统计信息是否由用户直接输入 (YES) 还是不 (NO)
DURATION	VARCHAR2 (15)	指示临时表的持续时间： <ul style="list-style-type: none"> • SYS\$SESSION-在会话期间保留行 • SYS\$TRANSACTION-行在以下时间后被删除COMMIT
PCT_DIRECT_ACCESS	VARCHAR2 (7)	对于索引组织表上的辅助索引，使用 VALID 猜测的行的百分比

名称	类型	描述
ITYP_OWNER	VARCHAR2(63)	对于域索引，索引类型的所有者
ITYP_NAME	VARCHAR2(63)	对于域索引，是索引类型的名称
PARAMETERS	VARCHAR2(1000)	对于域索引，参数字符串
GLOBAL_STATS	VARCHAR2(3)	如果收集或增量维护统计信息，则GLOBAL_STATS将为YES，否则将为NO
DOMIDX_STATUS	VARCHAR2(12)	域索引的状态： <ul style="list-style-type: none"> • 空索引不是域索引 • VALID- Index是有效的域索引 • IDXTYP_INVLD-域索引的 Indextype无效
DOMIDX_OPSTATUS	VARCHAR2(6)	域索引的操作状态： <ul style="list-style-type: none"> • 空索引不是域索引 • VALID-操作执行无误 • FAILED-操作失败，出现错误
FUNCIDX_STATUS	VARCHAR2(8)	基于函数的索引的状态： <ul style="list-style-type: none"> • 空索引不是基于函数的索引 • ENABLED-启用基于函数的索引 • DISABLED-基于函数的索引被禁用
JOIN_INDEX	TEXT	指示该索引是否是联接索引 (YES) 还是不 (NO)
IOT_REDUNDANT_PKEY_ELIM	TEXT	指示是否从按索引组织的表上的辅助索引中删除了冗余的主键列 (YES) 还是不 (NO)
DROPPED	TEXT	指示该索引是否已被删除并在回收站中 (YES) 还是不 (NO)；分区表为空。该视图不返回已被删除的索引的名称

1.104. all_objects

all_objects视图描述当前用户下可以查看的所有对象。

表 1.105. all_objects的列

名称	类型	描述
OWNER	VARCHAR2 (63)	对象的拥有者
OBJECT_NAME	VARCHAR2 (63)	对象名
SUBOBJECT_NAME	VARCHAR2 (63)	子对象的名称
OBJECT_ID	NUMERIC (38, 0)	对象的对象号
DATA_OBJECT_ID	NUMERIC (38, 0)	包含该对象的段的字典对象号
OBJECT_TYPE	VARCHAR2 (19)	对象类型
CREATED	DATE	对象创建的时间戳
LAST_DDL_TIME	DATE	对象和DDL语句产生的依赖对象的最后修改的时间戳(包括授予和撤销)
TIMESTAMP	VARCHAR2 (20)	对象规范的时间戳(字符数据)
STATUS	VARCHAR2 (7)	对象的状态
TEMPORARY	VARCHAR2 (1)	对象是否是临时的
GENERATED	VARCHAR2 (1)	是否生成此对象系统的名称
SECONDARY	VARCHAR2 (1)	指示这是否是Oracle Data Cartridge的ODCIIndexCreate方法创建的辅助对象
NAMESPACE	NUMERIC (38, 0)	命名空间

1.105. all_sequences

all_sequences视图描述当前用户可访问的所有序列。

表 1.106. all_sequences的列

名称	类型	描述
SEQUENCE_OWNER	VARCHAR2 (63)	序列的拥有者
SEQUENCE_NAME	VARCHAR2 (63)	序列名
MIN_VALUE	NUMERIC (38, 0)	序列的最小值
MAX_VALUE	NUMERIC (38, 0)	序列的最大值
INCREMENT_BY	NUMERIC (38, 0)	序列按该值递增
CYCLE_FLAG	TEXT	指示序列在达到极限(Y)时是否换行(N)
ORDER_FLAG	VARCHAR2 (1)	指示序列号是否按(Y)的顺序生成(N)
CACHE_SIZE	NUMERIC (38, 0)	要缓存的序列号的个数
LAST_NUMERIC	NUMERIC (38, 0)	最后写入磁盘的序列号。如果序列使用缓存, 则写入磁盘的数字是序列缓存中的最后一个数字。这个数字很可能大于所使用的最后一个序列号。对于会话序列, 应忽略此列中的值

1.106. all_source

all_source视图描述当前用户可访问的存储对象的文本源。

表 1.107. all_source的列

名称	类型	描述
OWNER	VARCHAR2 (63)	对象的拥有者
NAME	VARCHAR2 (63)	对象名
TYPE	VARCHAR2 (12)	对象类型:: FUNCTION, JAVA SOURCE, PACKAGE, PACKAGE BODY, PROCEDURE, TRIGGER, TYPE, TYPE BODY
TEXT	CLOB	存储对象的文本源

1.107. all_synonyms

描述当前用户所能查看的同义词信息。包括：

- 当前用户的私有同义词
- 所有的public同义词
- 本地同义词（DB_LINK列为空）所指向，或者嵌套指向的对象，当前用户有权限查看

表 1.108. all_synonyms的列

名称	类型	描述
OWNER	VARCHAR2 (63)	同义词的所有者
SYNONYM_NAMESPACE_NAME	VARCHAR2 (63)	同义词的命名空间
SYNONYM_NAME	VARCHAR2 (63)	同义词名
TABLE_OWNER	VARCHAR2 (63)	由同义词引用的对象的所有者
TABLE_NAMESPACE_NAME	VARCHAR2 (63)	表的命名空间
TABLE_NAME	VARCHAR2 (63)	由同义词引用的对象的名称
DBLINK	TEXT	引用的数据库链接的名称(如果有的话)

1.108. all_tab_cols

all_tab_cols视图描述当前用户可以访问的表，视图的列。

表 1.109. all_tab_cols的列

名称	类型	描述
OWNER	VARCHAR2 (63)	表、视图的拥有者
TABLE_NAME	VARCHAR2 (63)	表、视图的名称
COLUMN_NAME	VARCHAR2 (63)	列名
DATA_TYPE	VARCHAR2 (106)	列的数据类型

名称	类型	描述
DATA_TYPE_MOD	VARCHAR2(3)	列的数据类型修饰符
DATA_TYPE_OWNER	VARCHAR2(63)	列的数据类型的所有者
DATA_LENGTH	NUMERIC	列长度(以字节为单位)
DATA_PRECISION	NUMERIC	数字数据类型的十进制精度; 浮点型的二进制精度;所有其他数据类型为空
DATA_SCALE	NUMERIC	位数:数字中小数点右边的位数
NULLABLE	VARCHAR2(1)	指示列是否允许为空。如果列上存在非NULL约束或列是主键的一部分,则该值为N
COLUMN_ID	NUMERIC	创建的列的序列号
DEFAULT_LENGTH	NUMERIC	列的默认值的长度
DATA_DEFAULT	TEXT	列的默认值
NUM_DISTINCT	NUMERIC	列中不同值的数目
LOW_VALUE	NUMERIC	列中的低值
HIGH_VALUE	NUMERIC	列中的高值
DENSITY	NUMERIC	如果柱状图在COLUMN_NAME上可用,则此列显示在柱状图中跨越少于2个端点的值的选择性。它不表示跨越2个或多个端点的值的选择性。如果柱状图在COLUMN_NAME上不可用,则该列的值为1/NUM_DISTINCT
NUM_NULLS	NUMERIC	列中空值的数目
NUM_BUCKETS	NUMERIC	列的直方图中的桶数。注意:直方图中的桶数是在ANALYZE SQL语句的SIZE参数中指定的。然而,Oracle数据库并没有创建一个比样本行数更多的桶的直方图。此外,如果样本中包含任何重复的值,Oracle数据库会创建指定数量的桶,但由于内部压缩算法的原因,这一列显示的值可能更小
LAST_ANALYZED	TIMESTAMP(0) WITHOUT TIME ZONE	这一栏最近分析的日期
SAMPLE_SIZE	NUMERIC	用于分析该列的样本量
CHARACTER_SET_NAME	VARCHAR2(44)	字符集名称:CHAR_CS/NCHAR_CS
CHAR_COL_DECL_LENGTH	NUMERIC	字符类型列的声明长度
GLOBAL_STATS	VARCHAR2(3)	如果统计信息被收集或增量维护,GLOBAL_STATS将为YES,否则为NO
USER_STATS	VARCHAR2(3)	指示统计信息是否由用户直接输入(YES)或(NO)

名称	类型	描述
AVG_COL_LEN	NUMERIC	列的平均长度(以字节计)
CHAR_LENGTH	NUMERIC	以字符为单位显示列的长度。 该值仅适用于以下数据类型： <ul style="list-style-type: none"> • CHAR • VARCHAR2 • NCHAR • NVARCHAR2
CHAR_USED	VARCHAR2(1)	指示列使用BYTE长度语义(B)或CHAR长度语义(C)，或者数据类型不是以下任何一种(NULL)： <ul style="list-style-type: none"> • CHAR • VARCHAR2 • NCHAR • NVARCHAR2
V80_FMT_IMAGE	VARCHAR2(3)	指示列数据是否为8.0版本图像格式(YES)或否(NO)
DATA_UPGRADED	VARCHAR2(3)	指示列数据是否已升级为最新类型版本格式(YES)或(NO)
HIDDEN_COLUMN	VARCHAR2(3)	指示列是否是隐藏列(YES)或不是(NO)
VIRTUAL_COLUMN	VARCHAR2(3)	指示列是否是虚拟列(YES)或不是(NO)
SEGMENT_COLUMN_ID	NUMERIC	段中列的序列号
INTERNAL_COLUMN_ID	NUMERIC	列的内部序列号
HISTOGRAM	VARCHAR2(15)	表示直方图的存在/类型： <ul style="list-style-type: none"> • NONE • FREQUENCY • TOP-FREQUENCY • HEIGHT BALANCED • HYBRID
QUALIFIED_COL_NAME	VARCHAR2(4000)	限定的列名

1.109. all_tab_columns

all_tab_columns视图描述当前用户可以访问的表，视图的非隐藏列。

表 1.110. all_tab_columns的列

名称	类型	描述
OWNER	VARCHAR2(63)	表、视图的拥有者
TABLE_NAME	VARCHAR2(63)	表、视图的名称
COLUMN_NAME	VARCHAR2(63)	列名
DATA_TYPE	VARCHAR2(106)	列的数据类型
DATA_TYPE_MOD	VARCHAR2(3)	列的数据类型修饰符
DATA_TYPE_OWNER	VARCHAR2(63)	列的数据类型的所有者
DATA_LENGTH	NUMERIC	列长度(以字节为单位)
DATA_PRECISION	NUMERIC	NUMBER数据类型的十进制精度；FLOAT数据类型的二进制精度；所有其他数据类型为NULL
DATA_SCALE	NUMERIC	小数点右边的数字
NULLABLE	VARCHAR2(1)	指示列是否允许null。如果列上有NOT NULL约束，或者列是PRIMARY KEY的一部分，则该值为N
COLUMN_ID	NUMERIC	创建的列的序列号
DEFAULT_LENGTH	NUMERIC	列的默认值的长度
DATA_DEFAULT	TEXT	列的默认值
NUM_DISTINCT	NUMERIC	列中不同值的数目
LOW_VALUE	NUMERIC	列中的低值
HIGH_VALUE	NUMERIC	列中的高值
DENSITY	NUMERIC	如果柱状图在COLUMN_NAME上可用，则此列显示在柱状图中跨越少于2个端点的值的选择性。它不表示跨越2个或多个端点的值的选择性。如果柱状图在COLUMN_NAME上不可用，则该列的值为1/NUM_DISTINCT
NUM_NULLS	NUMERIC	列中空值的数目
NUM_BUCKETS	NUMERIC	列的直方图中的桶数。注意：直方图中的桶数是在ANALYZE SQL语句的SIZE参数中指定的。然而，Oracle数据库并没有创建一个比样本行数更多的桶的直方图。此外，如果样本中包含任何重复的值，Oracle数据库会创建指定数量的桶，但由于内部压缩算法的原因，这一列显示的值可能更小
LAST_ANALYZED	TIMESTAMP(0) WITHOUT TIME ZONE	这一栏最近分析的日期
SAMPLE_SIZE	NUMERIC	用于分析该列的样本量

名称	类型	描述
CHARACTER_SET_NAME	VARCHAR2 (44)	字符集名称: CHAR_CS/ NCHAR_CS
CHAR_COL_DECL_LENGTH	NUMERIC	字符类型列的声明长度
GLOBAL_STATS	VARCHAR2 (3)	如果统计信息被收集或增量维护, GLOBAL_STATS将为YES, 否则为NO
USER_STATS	VARCHAR2 (3)	指示统计信息是否由用户直接输入 (YES) 或 (NO)
AVG_COL_LEN	NUMERIC	列的平均长度 (以字节计)
CHAR_LENGTH	NUMERIC	以字符为单位显示列的长度。 该值仅适用于以下数据类型: <ul style="list-style-type: none"> • CHAR • VARCHAR2 • NCHAR • NVARCHAR2
CHAR_USED	VARCHAR2 (1)	指示列使用BYTE长度语义 (B) 或 CHAR长度语义 (C), 或者数据类型不是以下任何一种 (NULL): <ul style="list-style-type: none"> • CHAR • VARCHAR2 • NCHAR • NVARCHAR2
V80_FMT_IMAGE	VARCHAR2 (3)	指示列数据是否为8.0版本图像格式 (YES) 或否 (NO)
DATA_UPGRADED	VARCHAR2 (3)	指示列数据是否已升级为最新类型版本格式 (YES) 或 (NO)
HISTOGRAM	VARCHAR2 (15)	表示直方图的存在/类型: <ul style="list-style-type: none"> • NONE • FREQUENCY • TOP-FREQUENCY • HEIGHT BALANCED • HYBRID

1.110. all_tab_comments

all_tab_comments视图在当前用户可以访问的表和视图上显示注释。

表 1.111. all_tab_comments的列

名称	类型	描述
OWNER	VARCHAR2 (63)	对象的拥有者
TABLE_NAME	VARCHAR2 (63)	表名
TABLE_TYPE	VARCHAR2 (11)	表类型
COMMENTS	VARCHAR2 (4000)	注释

1.111. all_tab_privs

all_tab_privs视图描述当前用户，该用户可以是对象的所有者、授予者或者被授予者。

表 1.112. all_tab_privs的列

名称	类型	描述
GRANTEE	VARCHAR2 (63)	授予访问权的用户或角色的名称
OWNER	VARCHAR2 (63)	对象的所有者
TABLE_NAME	VARCHAR2 (63)	对象名称
GRANTOR	VARCHAR2 (63)	执行授权的用户的名称
PRIVILEGE	VARCHAR2 (40)	对象上的特权
GRANTABLE	VARCHAR2 (3)	指示是否使用GRANT OPTION (YES) 或否 (NO) 授予特权

1.112. all_tables

all_tables视图描述当前用户可访问的表。

表 1.113. all_tables 的列

名称	类型	描述
OWNER	VARCHAR2 (63)	表的拥有者
TABLE_NAME	VARCHAR2 (63)	表的名称
TABLESPACE_NAME	VARCHAR2 (63)	包含该表的表空间的名称；对于分区表、临时表和按索引组织的表，为NULL
CLUSTER_NAME	VARCHAR2 (63)	表所属的集群的名称(如果有)
IOT_NAME	VARCHAR2 (63)	溢出或映射表条目所属的按索引组织的表(如果有)的名称。如果IOT_TYPE列不为空，则此列包含基表名
STATUS	VARCHAR2 (63)	如果以前的DROP TABLE操作失败，指示表是否不可用 (UNUSABLE) 或有效 (VALID)
PCT_FREE	VARCHAR2 (63)	块中可用空间的最小百分比；分区表为空

名称	类型	描述
PCT_USED	INTEGER	块中已用空间的最小百分比；分区表为空
INI_TRANS	INTEGER	处理的初始数量；分区表为空
MAX_TRANS	INTEGER	最大处理数量；分区表为空
INITIAL_EXTENT	INTEGER	初始范围的大小(以字节为单位)；分区表为空
NEXT_EXTENT	INTEGER	二级区段的大小(以字节为单位)；NULL用于分区表
MIN_EXTENTS	INTEGER	段中允许的最小范围数；分区表为空
MAX_EXTENTS	INTEGER	段中允许的最大范围数；分区表为空
PCT_INCREASE	INTEGER	范围大小的增加百分比；分区表为空
FREELISTS	INTEGER	分配给该段的进程空闲列表数；分区表为空
FREELIST_GROUPS	INTEGER	分配给该段的空闲列表组的数量；分区表为空
LOGGING	INTEGER	指示是否记录对表的更改；分区表为空；YES/NO
BACKED_UP	VARCHAR2(3)	指示自最后一次修改(Y)以来是否备份了表(N)
NUM_ROWS	VARCHAR2(1)	表中的行数
BLOCKS	INTEGER	表中已使用数据块的个数
EMPTY_BLOCKS	INTEGER	表中空(从未使用过)数据块的数量。只有在使用DBMS_STATS包收集表上的统计信息时，才会填充此列
AVG_SPACE	INTEGER	分配给表的数据块中的平均可用空间量(以字节为单位)
CHAIN_CNT	INTEGER	表中从一个数据块链接到另一个数据块或已迁移到新块的行数，需要一个链接来保存旧的ROWID
AVG_ROW_LEN	INTEGER	表中一行的平均长度(以字节为单位)
AVG_SPACE_FREELIST_BLOCKS	INTEGER	自由列表中所有块的平均空闲空间
NUM_FREELIST_BLOCKS	INTEGER	自由列表上的块数
DEGREE	INTEGER	每个实例用于扫描表的线程数，或默认值
INSTANCES	VARCHAR2(10)	要扫描表的实例数，或默认值
CACHE	VARCHAR2(5)	指示是否将表缓存在缓冲区缓存(Y)中(N)

名称	类型	描述
TABLE_LOCK	VARCHAR2 (8)	表锁定是启用(enabled)还是禁用(disabled)
SAMPLE_SIZE	INTEGER	用于分析表格的样本量
LAST_ANALYZED	INTEGER	该表最近被分析的日期
PARTITIONED	DATE	指示表是否分区(YES)或未分区(NO)
IOT_TYPE	VARCHAR2 (3)	如果表是索引组织的表, 则 IOT_TYPE为IOT、IOT_OVERFLOW或IOT_MAPPING。如果表不是索引组织的表, 则IOT_TYPE为NULL
TEMPORARY	VARCHAR2 (12)	表是临时的(Y)还是非临时的(N)
SECONDARY	VARCHAR2 (1)	说明表是否为Oracle数据盒(Y)的ODCIIndexCreate方法创建的辅助对象(N)
NESTED	VARCHAR2 (1)	指示表是嵌套表(是)还是非嵌套表(否)
BUFFER_POOL	VARCHAR2 (3)	表的缓冲池; 分区表为空
ROW_MOVEMENT	VARCHAR2 (7)	指示分区行移动是启用(enabled)还是禁用(disabled)
GLOBAL_STATS	VARCHAR2 (8)	如果统计信息被收集或增量维护, GLOBAL_STATS将为YES, 否则为NO
USER_STATS	VARCHAR2 (3)	表示统计数据是由用户直接输入的(YES)还是不是(NO)
DURATION	VARCHAR2 (3)	临时表的持续时间, 取值如下所示。 <ul style="list-style-type: none"> • SYS\$SESSION - 行在会话期间保留 • SYS\$TRANSACTION - 行在提交后被删除 • Null - 永久表
SKIP_CORRUPT	VARCHAR2 (15)	指示Oracle数据库是否在表和索引扫描期间忽略标记为损坏的块(启用)或引发错误(禁用)。要启用此特性, 请运行DBMS_REPAIR.SKIP_CORRUPT_BLOCKS过程
MONITORING	VARCHAR2 (8)	死元组
CLUSTER_OWNER	VARCHAR2 (3)	表所属集群的所有者(如果有的话)

名称	类型	描述
DEPENDENCIES	VARCHAR2 (63)	指示行级依赖项跟踪是启用(启用)还是禁用(禁用)
COMPRESSION	VARCHAR2 (8)	表压缩是否启用(启用)或不启用(禁用);对于分区表, 为NULL
DROPPED	VARCHAR2 (8)	指示表是否已被删除, 是否在回收站(YES)或(NO);对于分区表, 为NULL, 此视图不返回已删除的表的名称
varcharbyte	VARCHAR2 (3)	默认值

1.113. all_trigger_cols

all_trigger_cols视图描述当前用户可访问的所有触发器的列。

表 1.114. all_trigger_cols的列

名称	类型	描述
TRIGGER_OWNER	VARCHAR2 (63)	触发器的所有者
TRIGGER_NAME	VARCHAR2 (63)	触发器名
TABLE_OWNER	VARCHAR2 (63)	表的所有者
TABLE_NAME	VARCHAR2 (63)	表名
COLUMN_NAME	VARCHAR2 (4000)	列名

1.114. all_triggers

all_triggers视图描述当前用户可访问的所有触发器。

表 1.115. all_triggers的列

名称	类型	描述
OWNER	VARCHAR2 (63)	对象的所有者
TRIGGER_NAME	VARCHAR2 (63)	触发器名
TRIGGER_TYPE	VARCHAR2 (16)	触发器类型
TRIGGERING_EVENT	VARCHAR2 (216)	触发触发器的DML、DDL或数据库事件
TABLE_OWNER	VARCHAR2 (63)	表的所有者
TABLE_NAME	VARCHAR2 (63)	表名
BASE_OBJECT_TYPE	VARCHAR2 (16)	定义触发器的基对象, 取值如下所示 <ul style="list-style-type: none"> • TABLE • VIEW • SCHEMA

名称	类型	描述
		• DATABASE
COLUMN_NAME	VARCHAR2 (4000)	列名
REFERENCING_NAMES	VARCHAR2 (422)	用于从触发器中引用OLD和NEW列值的名称
DESCRIPTION	VARCHAR2 (4000)	触发描述；用于重新创建触发器创建语句
STATUS	VARCHAR2 (8)	指示触发器是启用(enabled)还是禁用(disabled)；禁用的触发器不会触发
TRIGGER_BODY	TEXT	触发器触发时执行的语句

1.115. all_users

all_users视图列出当前用户可见的数据库的所有用户。

表 1.116. all_users的列

名称	类型	描述
USERNAME	VARCHAR2 (63)	用户名
USER_ID	NUMERIC (38, 0)	用户id
CREATED	DATE	创建用户的时间

1.116. all_views

all_views视图描述当前用户所能查看的所有的视图信息。

表 1.117. all_views的列

名称	类型	描述
OWNER	VARCHAR2 (63)	视图的拥有者
VIEW_NAME	VARCHAR2 (63)	视图名
TEXT_LENGTH	NUMERIC	视图文本的长度
TEXT	TEXT	查看文本。只有当行起源于当前容器时，此列才返回正确的值。在此视图中，BEQUEATH子句不会作为TEXT列的一部分出现
TYPE_TEXT_LENGTH	NUMERIC	视图的类型子句的长度
TYPE_TEXT	VARCHAR2 (4000)	视图的类型子句
OID_TEXT_LENGTH	NUMERIC	视图的WITH OID子句的长度
OID_TEXT	VARCHAR2 (4000)	视图的WITH OID子句
VIEW_TYPE_OWNER	VARCHAR2 (63)	如果视图是类型化视图，则为视图类型的所有者
VIEW_TYPE	VARCHAR2 (63)	如果视图是类型化视图，则为视图的类型

名称	类型	描述
SUPERVIEW_NAME	VARCHAR2(63)	父视图的名称
EDITIONING_VIEW	VARCHAR2(1)	保留供将来使用
READ_ONLY	VARCHAR2(1)	指示视图是否只读(Y) (N)

1.117. db_files

db_files视图描述数据库中所有表空间、表空间使用的存储容量及表空间所在存储设备的容量信息。

表 1.118. db_files的列

名称	类型	描述
NAME	NAME	命名空间
USED	BIGINT	使用的百分比
CAPACITY	BIGINT	总空间
RATIO	NUMERIC	百分比

1.118. dba_arguments

dba_arguments视图列出数据库中可用的过程和函数的参数。

表 1.119. dba_arguments的列

名称	类型	描述
OWNER	CHARACTER VARYING	对象的拥有者
PACKAGE_NAME	CHARACTER VARYING	包的名称
OBJECT_NAME	CHARACTER VARYING	过程或函数的名称
OBJECT_ID	OID	对象的对象号
POSITION	INTEGER	此列保存该项在参数列表中的位置，0表示函数返回值
ARGUMENT_NAME	CHARACTER VARYING	参数名称，空参数名用于表示函数返回
IN_OUT	CHARACTER VARYING	入参/出参类型 <ul style="list-style-type: none"> • IN • OUT • IN/OUT
TYPE_NAME	CHARACTER VARYING	实参类型的名称。如果类型是包本地类型(也就是说，它在包规范中声明)，则此列显示包的名称
OVERLOAD	CHARACTER VARYING	指示第n个重载(按其在源中的出现顺序)；否则，它是NULL
SUBPROGRAM_ID	INTEGER	唯一子程序标识符

名称	类型	描述
SEQUENCE	INTEGER	定义参数的顺序。参数序列从1开始。首先是返回类型，然后是每个参数
DATA_LEVEL	INTEGER	复合类型实参的嵌套深度
DATA_TYPE	CHARACTER VARYING	参数的数据类型
DEFAULTED	CHARACTER VARYING	指定参数是否为默认值
DEFAULT_VALUE	TEXT	保留供将来使用
DEFAULT_LENGTH	INTEGER	保留供将来使用
DATA_LENGTH	INTEGER	列长度(以字节为单位)
DATA_PRECISION	INTEGER	十进制数字(NUMBER)或二进制数字(FLOAT)的长度
DATA_SCALE	INTEGER	小数点右边的数字
RADIX	INTEGER	一个数字的基数参数
CHARACTER_SET_NAME	CHARACTER VARYING	参数的字符集名称
TYPE_OWNER	CHARACTER VARYING	参数类型的所有者
TYPE_SUBNAME	CHARACTER VARYING	只与包本地类型相关。显示在TYPE_NAME列标识的包中声明的类型的名称
TYPE_LINK	CHARACTER VARYING	当TYPE_NAME列中标识的包是远程包时，仅与包本地类型相关。此列显示用于引用远程包的数据库链接
TYPE_OBJECT_TYPE	CHARACTER VARYING	显示由TYPE_OWNER、TYPE_NAME和TYPE_SUBNAME列描述的类型。取值包括： <ul style="list-style-type: none"> • TABLE • VIEW • PACKAGE • TYPE
PLS_TYPE	CHARACTER VARYING	对于数值形参，实参的PL/SQL类型的名称。否则无效
CHAR_LENGTH	INTEGER	字符串数据类型的字符限制
CHAR_USED	CHARACTER VARYING	指示字符串的字节限制(B)或字符限制(C)是否正式
ORIGIN_CON_ID	INTEGER	数据产生的容器的ID。可能的值包括： <ul style="list-style-type: none"> • 0：该值用于非cdb中的行。此值不用于cdb • n：该值用于包含起源于容器ID为n的容器的数据的行(如

名称	类型	描述
		果该行起源于root，则n = 1)

1.119. dba_col_comments

dba_col_comments视图在数据库中所有表和视图的列上显示注释。

表 1.120. dba_col_comments的列

名称	类型	描述
OWNER	VARCHAR2 (63)	对象的拥有者
TABLE_NAME	VARCHAR2 (63)	对象名
COLUMN_NAME	VARCHAR2 (63)	列名
COMMENTS	VARCHAR2 (4000)	描述

1.120. dba_col_privs

dba_col_privs视图描述所有列级权限。

表 1.121. dba_col_privs的列

名称	类型	描述
GRANTOR	VARCHAR2 (63)	执行授权的用户的名称
OWNER	VARCHAR2 (63)	对象的拥有者
GRANTEE	VARCHAR2 (63)	授予访问权的用户或角色的名称
TABLE_CATALOG	VARCHAR2 (63)	当前的数据库
TABLE_SCHEMA	VARCHAR2 (63)	对象的模式
TABLE_NAME	VARCHAR2 (63)	表名
COLUMN_NAME	VARCHAR2 (63)	列名
PRIVILEGE_TYPE	VARCHAR2 (40)	权限名称
IS_GRANTABLE	VARCHAR2 (3)	允许访问

1.121. dba_cons_columns

dba_cons_columns视图描述约束中指定的数据库中的所有列。

表 1.122. dba_cons_columns的列

名称	类型	描述
OWNER	VARCHAR2 (63)	兑现的拥有者
CONSTRAINT_NAME	VARCHAR2 (63)	约束名
TABLE_NAME	VARCHAR2 (63)	表名

名称	类型	描述
COLUMN_NAME	VARCHAR2 (4000)	列名
POSITION	NUMERIC	定义中列的原始位置

1.122. dba_constraints

dba_constraints视图描述当前用户所拥有的表的所有约束定义。

表 1.123. dba_constraints的列

名称	类型	描述
OWNER	VARCHAR2 (63)	对象的拥有者
CONSTRAINT_NAME	VARCHAR2 (63)	约束名
CONSTRAINT_TYPE	VARCHAR2 (1)	约束定义的类型，如下所示。 <ul style="list-style-type: none"> • C -表上的检查约束 • P -主键 • U -唯一密钥 • R -外键 • T -受限触发器 • X -排他性约束
TABLE_NAME	VARCHAR2 (63)	表名
SEARCH_CONDITION	TEXT	检查约束的搜索条件的文本。只有当行起源于当前容器时，此列才返回正确的值
R_OWNER	VARCHAR2 (63)	引用约束中引用的表的所有者
R_CONSTRAINT_NAME	VARCHAR2 (63)	引用表的唯一约束定义的名称
DELETE_RULE	VARCHAR2 (9)	引用约束的删除规则，如下所示。 <ul style="list-style-type: none"> • RESTRICT • CASCADE • NULL • SET NULL • SET DEFAULT • NO ACTION
STATUS	TEXT	约束的实施状态，如下所示。 <ul style="list-style-type: none"> • ENABLED • DISABLED

名称	类型	描述
DEFERRABLE	VARCHAR2(14)	指示约束是否可延迟(deferrable)或不可延迟(not deferrable)
DEFERRED	VARCHAR2(9)	指示约束最初是否被延迟(deferred) (IMMEDIATE)
VALIDATED	VARCHAR2(13)	<p>当STATUS = ENABLED时, 可能的取值如下所示。</p> <ul style="list-style-type: none"> VALIDATED -所有数据都遵守约束(也就是说, 当启用约束时, 表中的现有数据以及输入到表中的任何后续数据都将得到验证) NOVALIDATED -所有数据都可能不遵守约束(也就是说, 当启用约束时, 表中现有的数据没有被验证, 但是输入到表中的后续数据被验证)
GENERATED	VARCHAR2(14)	指示约束的名称是用户生成(USER name)还是系统生成(GENERATED name)
BAD	VARCHAR2(3)	指示此约束是否以模糊的方式指定世纪(BAD)或否(NULL)。为了避免这种模糊性导致的错误, 可以使用带四位数字年份的TO_DATE函数重写约束
RELY	VARCHAR2(4)	当VALIDATED = NOT VALIDATED时, 此列表示查询重写(RELY)时是否考虑约束(NULL)
LAST_CHANGE	TIMESTAMP(0) WITHOUT TIME ZONE	约束最后一次启用或禁用的时间
INDEX_OWNER	VARCHAR2(63)	拥有索引的用户名
INDEX_NAME	VARCHAR2(63)	索引的名称(仅在唯一和主键约束时显示)
INVALID	VARCHAR2(7)	指示约束是否无效(invalid)或无效(NULL)
VIEW_RELATED	VARCHAR2(14)	指示约束是否依赖于视图(DEPEND on view) (NULL)

1.123. dba_ind_columns

dba_ind_columns视图描述数据库中所有建有索引的列的信息。

表 1.124. dba_ind_columns的列

名称	类型	描述
INDEX_OWNER	VARCHAR2(63)	对象的拥有者

名称	类型	描述
INDEX_NAME	VARCHAR2 (63)	索引名
TABLE_OWNER	VARCHAR2 (63)	表的拥有者
TABLE_NAME	VARCHAR2 (63)	表名
COLUMN_NAME	VARCHAR2 (4000)	列名
COLUMN_POSITION	SMALLINT	索引中的列或属性的位置
COLUMN_LENGTH	NUMERIC	列的索引长度
CHAR_LENGTH	NUMERIC	列的最大码点长度
DESCEND	VARCHAR2 (4)	列是否按降序排序 (Y/N)

1.124. dba_indexes

dba_indexes视图描述数据库中所有索引的信息。

表 1.125. dba_indexes的列

名称	类型	描述
OWNER	VARCHAR2 (63)	索引的拥有者
INDEX_NAME	VARCHAR2 (63)	索引名
INDEX_TYPE	VARCHAR2 (63)	索引类型
TABLE_OWNER	VARCHAR2 (63)	表的拥有者
TABLE_NAME	VARCHAR2 (63)	表名
TABLE_TYPE	TEXT	表类型
UNIQUENESS	VARCHAR2 (9)	索引的惟一状态
COMPRESSION	VARCHAR2 (8)	用于索引的压缩类型
PREFIX_LENGTH	NUMERIC	压缩键前缀中的列数
TABLESPACE_NAME	VARCHAR2 (63)	包含索引的表空间的名称
INI_TRANS	VARCHAR2 (7)	初始交易数量
MAX_TRANS	VARCHAR2 (7)	最大交易数量
INITIAL_EXTENT	VARCHAR2 (7)	初始范围的大小
NEXT_EXTENT	VARCHAR2 (7)	辅助扩展区的大小
MIN_EXTENTS	VARCHAR2 (7)	段中允许的最小范围数
MAX_EXTENTS	VARCHAR2 (7)	段中允许的最大范围数
PCT_INCREASE	VARCHAR2 (7)	区大小增加的百分比
PCT_THRESHOLD	VARCHAR2 (7)	每个索引条目允许的块空间百分比阈值
INCLUDE_COLUMN	VARCHAR2 (7)	要包含在按索引组织的表主键 (非溢出) 索引中的最后一列的列ID
FREELISTS	VARCHAR2 (7)	分配给该段的进程空闲列表数
FREELIST_GROUPS	VARCHAR2 (7)	分配给该段的空闲列表组的数量

名称	类型	描述
PCT_FREE	VARCHAR2 (7)	块中可用空间的最小百分比
LOGGING	VARCHAR2 (7)	指示是否记录对索引的更改:
BLEVEL	VARCHAR2 (7)	b *-树级别 (从其根块到其叶块的索引深度)。深度0表示根块和叶块是相同的
LEAF_BLOCKS	VARCHAR2 (7)	索引中的叶块数
DISTINCT_KEYS	VARCHAR2 (7)	不同索引值的数量。对于强制执行的索引UNIQUE和PRIMARY KEY约束, 该值与表中的行数相同(*_TABLES.NUM_ROWS)
AVG_LEAF_BLOCKS_PER_KEY	VARCHAR2 (7)	索引中每个不同值出现的叶块数的平均值, 四舍五入为最接近的整数。对于强制执行的索引UNIQUE和PRIMARY KEY约束, 该值始终为1
AVG_DATA_BLOCKS_PER_KEY	VARCHAR2 (7)	表中由索引中的非重复值指向的平均数据块数, 四舍五入为最接近的整数。此统计信息是包含索引列中包含给定值的行的数据块的平均数量
CLUSTERING_FACTOR	VARCHAR2 (7)	根据索引值指示表中行的顺序 如果该值接近块数, 则该表非常有序。在这种情况下, 单个叶块中的索引条目倾向于指向相同数据块中的行 如果该值接近行数, 则该表是随机排序的。在这种情况下, 同一叶块中的索引条目不太可能指向同一数据块中的行
STATUS	VARCHAR2 (8)	指示非分区索引是否为VALID或者UNUSABLE
NUM_ROWS	VARCHAR2 (7)	索引中的行数。
SAMPLE_SIZE	VARCHAR2 (7)	用于分析指数的样本大小
LAST_ANALYZED	VARCHAR2 (20)	最近分析此索引的日期
DEGREE	VARCHAR2 (40)	每个实例用于扫描索引的线程数, 或者DEFAULT
INSTANCES	VARCHAR2 (40)	要扫描索引的实例数, 或者DEFAULT
PARTITIONED	VARCHAR2 (3)	指示索引是否已分区 (YES) 还是不 (NO)
TEMPORARY	VARCHAR2 (1)	指示索引是否在临时表上 (Y) 还是不 (N)
GENERATED	VARCHAR2 (1)	指示索引的名称是否是系统生成的 (Y) 还是不 (N)

名称	类型	描述
SECONDARY	VARCHAR2(1)	指示该索引是否是由 ODCIIndexCreateOracle数据盒 式磁带的方法(Y)还是不(N)
BUFFER_POOL	VARCHAR2(7)	用于索引块的缓冲池:
USER_STATS	VARCHAR2(3)	指示统计信息是否由用户直接 输入(YES)还是不(NO)
DURATION	VARCHAR2(15)	指示临时表的持续时间 <ul style="list-style-type: none"> • SYS\$SESSION-在会话期间保留行 • SYS\$TRANSACTION-行在以下 时间后被删除COMMIT
PCT_DIRECT_ACCESS	VARCHAR2(7)	对于按索引组织的表上的辅助 索引, 包含以下内容的行的百 分比VALID猜测
ITYP_OWNER	VARCHAR2(63)	对于域索引, 索引类型的所有 者
ITYP_NAME	VARCHAR2(63)	对于域索引, 是索引类型的名 称
PARAMETERS	VARCHAR2(1000)	对于域索引, 参数字符串
GLOBAL_STATS	VARCHAR2(3)	GLOBAL_STATS将会YES如果统计 数据被收集或增量维护, 则它 将NO
DOMIDX_STATUS	VARCHAR2(12)	域索引的状态, 如下所示 <ul style="list-style-type: none"> • 空索引不是域索引 • VALID- Index是有效的域索 引 • IDXTYP_INVLD-域索引的 Indextype无效
DOMIDX_OPSTATUS	VARCHAR2(6)	域索引的操作状态, 如下所示 <ul style="list-style-type: none"> • 空索引不是域索引 • VALID-操作执行无误 • FAILED-操作失败, 出现错误
FUNCIDX_STATUS	VARCHAR2(8)	基于函数的索引的状态, 如下 所示 <ul style="list-style-type: none"> • 空索引不是基于函数的索引 • ENABLED-启用基于函数的索 引

名称	类型	描述
		<ul style="list-style-type: none"> DISABLED-基于函数的索引被禁用
JOIN_INDEX	TEXT	指示该索引是否是联接索引 (YES) 还是不 (NO)
IOT_REDUNDANT_PKEY_ELIM	TEXT	指示是否从按索引组织的表上的辅助索引中删除了冗余的主键列 (YES) 还是不 (NO)
DROPPED	TEXT	指示该索引是否已被删除并在回收站中 (YES) 还是不 (NO)；分区表为空。该视图不返回已被删除的索引的名称

1.125. dba_objects

dba_objects视图描述当前用户下可以查看的所有对象。

表 1.126. dba_objects的列

名称	类型	描述
OWNER	VARCHAR2 (63)	对象的拥有者
OBJECT_NAME	VARCHAR2 (63)	对象名
SUBOBJECT_NAME	VARCHAR2 (63)	子对象的名称
OBJECT_ID	NUMERIC (38, 0)	对象的对象号
DATA_OBJECT_ID	NUMERIC (38, 0)	包含该对象的段的字典对象号
OBJECT_TYPE	VARCHAR2 (19)	对象类型
CREATED	DATE	对象创建的时间戳
LAST_DDL_TIME	DATE	对象和DDL语句产生的依赖对象的最后修改的时间戳(包括授予和撤销)
TIMESTAMP	VARCHAR2 (20)	对象规范的时间戳(字符数据)
STATUS	VARCHAR2 (7)	对象的状态
TEMPORARY	VARCHAR2 (1)	对象是否是临时的
GENERATED	VARCHAR2 (1)	是否生成此对象系统的名称
SECONDARY	VARCHAR2 (1)	指示这是否是Oracle Data Cartridge的ODCIIndexCreate方法创建的辅助对象
NAMESPACE	NUMERIC (38, 0)	命名空间

1.126. dba_role_privs

dba_role_privs视图描述授予所有用户的角色以及数据库中的角色。

表 1.127. dba_role_privs的列

名称	类型	描述
GRANTEE	VARCHAR2 (63)	接收授权的用户或角色的名称
GRANTED_ROLE	VARCHAR2 (63)	授予的角色名
ADMIN_OPTION	VARCHAR2 (3)	指示授予是否带有管理选项 (YES) 或否 (NO)
DEFAULT_ROLE	VARCHAR2 (3)	指示角色是否被指定为用户的默认角色 (YES) 或否 (NO)

1.127. dba_roles

dba_roles视图列出数据库中存在的角色。

表 1.128. dba_roles的列

名称	类型	描述
ROLE	VARCHAR2 (63)	角色名称
PASSWORD_REQUIRED	VARCHAR2 (28)	密码

1.128. dba_sequences

dba_sequences视图描述数据库中所有的序列。

表 1.129. dba_sequences的列

名称	类型	描述
SEQUENCE_OWNER	VARCHAR2 (63)	序列的拥有者
SEQUENCE_NAME	VARCHAR2 (63)	序列名
MIN_VALUE	NUMERIC (38, 0)	序列的最小值
MAX_VALUE	NUMERIC (38, 0)	序列的最大值
INCREMENT_BY	NUMERIC (38, 0)	序列按该值递增
CYCLE_FLAG	VARCHAR2 (1)	指示序列在达到极限 (Y) 时是否换行 (N)
ORDER_FLAG	VARCHAR2 (1)	指示序列号是否按 (Y) 的顺序生成 (N)
CACHE_SIZE	NUMERIC (38, 0)	要缓存的序列号的个数
LAST_NUMERIC	NUMERIC (38, 0)	最后写入磁盘的序列号。如果序列使用缓存，则写入磁盘的数字是序列缓存中的最后一个数字。这个数字很可能大于所使用的最后一个序列号。对于会话序列，应忽略此列中的值

1.129. dba_source

dba_source视图描述数据库中所有存储对象的文本源。

表 1.130. dba_source的列

名称	类型	描述
OWNER	VARCHAR2 (63)	对象的拥有者
NAME	VARCHAR2 (63)	对象名
TYPE	VARCHAR2 (12)	对象类型:: FUNCTION, JAVA SOURCE, PACKAGE, PACKAGE BODY, PROCEDURE, TRIGGER, TYPE, TYPE BODY
TEXT	VARCHAR2 (4000)	存储对象的文本源

1.130. dba_synonyms

dba_synonyms视图描述数据库中所有同义词的信息。

表 1.131. dba_synonyms的列

名称	类型	描述
OWNER	VARCHAR2 (63)	同义词的所有者
SYNONYM_NAMESPACE_NAME	VARCHAR2 (63)	同义词的命名空间
SYNONYM_NAME	VARCHAR2 (63)	同义词名
TABLE_OWNER	VARCHAR2 (63)	由同义词引用的对象的所有者
TABLE_NAMESPACE_NAME	VARCHAR2 (63)	表的命名空间
TABLE_NAME	VARCHAR2 (63)	由同义词引用的对象的名称
DBLINK	TEXT	引用的数据库链接的名称(如果有的话)

1.131. dba_tab_cols

dba_tab_cols视图描述当前所在数据库中，所有的表、视图的列信息。

表 1.132. dba_tab_cols 的列

名称	类型	描述
OWNER	VARCHAR2 (63)	表、视图的拥有者
TABLE_NAME	VARCHAR2 (63)	表、视图的名称
COLUMN_NAME	VARCHAR2 (63)	列名
DATA_TYPE	VARCHAR2 (106)	列的数据类型
DATA_TYPE_MOD	VARCHAR2 (3)	列的数据类型修饰符
DATA_TYPE_OWNER	VARCHAR2 (63)	列的数据类型的所有者
DATA_LENGTH	NUMERIC	列长度(以字节为单位)
DATA_PRECISION	NUMERIC	数字数据类型的十进制精度;浮点型的二进制精度;所有其他数据类型为空
DATA_SCALE	NUMERIC	位数:数字中小数点右边的位数

名称	类型	描述
NULLABLE	VARCHAR2(1)	指示列是否允许为空。如果列上存在非NULL约束或列是主键的一部分，则该值为N
COLUMN_ID	NUMERIC	创建的列的序列号
DEFAULT_LENGTH	NUMERIC	列的默认值的长度
DATA_DEFAULT	text	列的默认值
NUM_DISTINCT	NUMERIC	列中不同值的数目
LOW_VALUE	NUMERIC	列中的低值
HIGH_VALUE	NUMERIC	列中的高值
DENSITY	NUMERIC	如果柱状图在COLUMN_NAME上可用，则此列显示在柱状图中跨越少于2个端点的值的选择性。它不表示跨越2个或多个端点的值的选择性。如果柱状图在COLUMN_NAME上不可用，则该列的值为1/NUM_DISTINCT
NUM_NULLS	NUMERIC	列中空值的数目
NUM_BUCKETS	NUMERIC	列的直方图中的桶数。注意：直方图中的桶数是在ANALYZE SQL语句的SIZE参数中指定的。然而，Oracle数据库并没有创建一个比样本行数更多的桶的直方图。此外，如果样本中包含任何重复的值，Oracle数据库会创建指定数量的桶，但由于内部压缩算法的原因，这一列显示的值可能更小
LAST_ANALYZED	TIMESTAMP(0) WITHOUT TIME ZONE	这一栏最近分析的日期
SAMPLE_SIZE	NUMERIC	用于分析该列的样本量
CHARACTER_SET_NAME	VARCHAR2(44)	字符集名称:CHAR_CS/NCHAR_CS
CHAR_COL_DECL_LENGTH	NUMERIC	字符类型列的声明长度
GLOBAL_STATS	VARCHAR2(3)	如果统计信息被收集或增量维护，GLOBAL_STATS将为YES，否则为NO
USER_STATS	VARCHAR2(3)	指示统计信息是否由用户直接输入(YES)或(NO)
AVG_COL_LEN	NUMERIC	列的平均长度(以字节计)
CHAR_LENGTH	NUMERIC	以字符为单位显示列的长度。该值仅适用于以下数据类型。 <ul style="list-style-type: none"> CHAR VARCHAR2 NCHAR

名称	类型	描述
		<ul style="list-style-type: none"> NVARCHAR2
CHAR_USED	VARCHAR2(1)	指示列使用BYTE长度语义(B)或CHAR长度语义(C)，或者数据类型不是以下任何一种(NULL)。 <ul style="list-style-type: none"> CHAR VARCHAR2 NCHAR NVARCHAR2
V80_FMT_IMAGE	VARCHAR2(3)	指示列数据是否为8.0版本图像格式(YES)或否(NO)
DATA_UPGRADED	VARCHAR2(3)	指示列数据是否已升级为最新类型版本格式(YES)或(NO)
HIDDEN_COLUMN	VARCHAR2(3)	指示列是否是隐藏列(YES)或不是(NO)
VIRTUAL_COLUMN	VARCHAR2(3)	指示列是否是虚拟列(YES)或不是(NO)
SEGMENT_COLUMN_ID	NUMERIC	段中列的序列号
INTERNAL_COLUMN_ID	NUMERIC	列的内部序列号
HISTOGRAM	VARCHAR2(15)	表示直方图的存在/类型。取值如下所示。 <ul style="list-style-type: none"> NONE FREQUENCY TOP-FREQUENCY HEIGHT BALANCED HYBRID
QUALIFIED_COL_NAME	VARCHAR2(4000)	限定的列名

1.132. dba_tab_columns

dba_tab_columns视图描述当前所在数据库中，所有的表、视图的非隐藏列信息。

表 1.133. dba_tab_columns 的列

名称	类型	描述
OWNER	VARCHAR2(63)	表的拥有者
TABLE_NAME	VARCHAR2(63)	表、视图的名称
COLUMN_NAME	VARCHAR2(63)	列名
DATA_TYPE	VARCHAR2(106)	列的数据类型
DATA_TYPE_MOD	VARCHAR2(3)	列的数据类型修饰符

名称	类型	描述
DATA_TYPE_OWNER	VARCHAR2(63)	列的数据类型的所有者
DATA_LENGTH	NUMERIC	列长度(以字节为单位)
DATA_PRECISION	NUMERIC	数字数据类型的十进制精度；浮点型的二进制精度；所有其他数据类型为空
DATA_SCALE	NUMERIC	位数:数字中小数点右边的位数
NULLABLE	VARCHAR2(1)	指示列是否允许为空。如果列上存在非NULL约束或列是主键的一部分，则该值为N
COLUMN_ID	NUMERIC	创建的列的序列号
DEFAULT_LENGTH	NUMERIC	列的默认值的长度
DATA_DEFAULT	text	列的默认值
NUM_DISTINCT	NUMERIC	列中不同值的数目
LOW_VALUE	NUMERIC	列中的低值
HIGH_VALUE	NUMERIC	列中的高值
DENSITY	NUMERIC	如果柱状图在COLUMN_NAME上可用，则此列显示在柱状图中跨越少于2个端点的值的选择性。它不表示跨越2个或多个端点的值的选择性。如果柱状图在COLUMN_NAME上不可用，则该列的值为1/NUM_DISTINCT
NUM_NULLS	NUMERIC	列中空值的数目
NUM_BUCKETS	NUMERIC	列的直方图中的桶数。注意：直方图中的桶数是在ANALYZE SQL语句的SIZE参数中指定的。然而，Oracle数据库并没有创建一个比样本行数更多的桶的直方图。此外，如果样本中包含任何重复的值，Oracle数据库会创建指定数量的桶，但由于内部压缩算法的原因，这一列显示的值可能更小
LAST_ANALYZED	TIMESTAMP(0) WITHOUT TIME ZONE	这一栏最近分析的日期
SAMPLE_SIZE	NUMERIC	用于分析该列的样本量
CHARACTER_SET_NAME	VARCHAR2(44)	字符集名称:CHAR_CS/NCHAR_CS
CHAR_COL_DECL_LENGTH	NUMERIC	字符类型列的声明长度
GLOBAL_STATS	VARCHAR2(3)	如果统计信息被收集或增量维护，GLOBAL_STATS将为YES，否则为NO
USER_STATS	VARCHAR2(3)	指示统计信息是否由用户直接输入(YES)或(NO)
AVG_COL_LEN	NUMERIC	列的平均长度(以字节计)

名称	类型	描述
CHAR_LENGTH	NUMERIC	以字符为单位显示列的长度。 该值仅适用于以下数据类型。 <ul style="list-style-type: none"> • CHAR • VARCHAR2 • NCHAR • NVARCHAR2
CHAR_USED	VARCHAR2 (1)	指示列使用BYTE长度语义(B)或CHAR长度语义(C)，或者数据类型不是以下任何一种(NULL)。 <ul style="list-style-type: none"> • CHAR • VARCHAR2 • NCHAR • NVARCHAR2
V80_FMT_IMAGE	VARCHAR2 (3)	指示列数据是否为8.0版本图像格式(YES)或否(NO)
DATA_UPGRADED	VARCHAR2 (3)	指示列数据是否已升级为最新类型版本格式(YES)或(NO)
HISTOGRAM	VARCHAR2 (15)	表示直方图的存在/类型。取值如下所示。 <ul style="list-style-type: none"> • NONE • FREQUENCY • TOP-FREQUENCY • HEIGHT BALANCED • HYBRID

1.133. dba_tab_comments

dba_tab_comments视图显示数据库中所有表和视图的注释。

表 1.134. dba_tab_comments的列

名称	类型	描述
OWNER	VARCHAR2 (63)	对象的拥有者
TABLE_NAME	VARCHAR2 (63)	表名
TABLE_TYPE	VARCHAR2 (11)	表类型
COMMENTS	VARCHAR2 (4000)	注释

1.134. dba_tab_privs

dba_tab_privs视图描述数据库中的所有对象授予。

表 1.135. dba_tab_privs的列

名称	类型	描述
GRANTEE	VARCHAR2 (63)	授予访问权的用户或角色的名称
OWNER	VARCHAR2 (63)	对象的所有者
TABLE_NAME	VARCHAR2 (63)	对象名称
GRANTOR	VARCHAR2 (63)	执行授权的用户的名称
PRIVILEGE	VARCHAR2 (40)	对象上的特权
GRANTABLE	VARCHAR2 (3)	指示是否使用GRANT OPTION (YES) 或否 (NO) 授予特权

1.135. dba_tables

dba_tables视图描述数据库中的所有表。

表 1.136. dba_tables 的列

名称	类型	描述
OWNER	VARCHAR2 (63)	表的拥有者
TABLE_NAME	VARCHAR2 (63)	表的名称
TABLESPACE_NAME	VARCHAR2 (63)	包含该表的表空间的名称；对于分区表、临时表和按索引组织的表，为NULL
CLUSTER_NAME	VARCHAR2 (63)	表所属的集群的名称(如果有)
IOT_NAME	VARCHAR2 (63)	溢出或映射表条目所属的按索引组织的表(如果有)的名称。如果IOT_TYPE列不为空，则此列包含基表名
STATUS	VARCHAR2 (63)	如果以前的DROP TABLE操作失败，指示表是否不可用 (UNUSABLE) 或有效 (VALID)
PCT_FREE	INTEGER	块中可用空间的最小百分比；分区表为空
PCT_USED	INTEGER	块中已用空间的最小百分比；分区表为空
INI_TRANS	INTEGER	处理的初始数量；分区表为空
MAX_TRANS	INTEGER	最大处理数量；分区表为空
INITIAL_EXTENT	INTEGER	初始范围的大小(以字节为单位)；分区表为空
NEXT_EXTENT	INTEGER	二级区段的大小(以字节为单位)；NULL用于分区表

名称	类型	描述
MIN_EXTENTS	INTEGER	段中允许的最小范围数；分区表为空
MAX_EXTENTS	INTEGER	段中允许的最大范围数；分区表为空
PCT_INCREASE	INTEGER	范围大小的增加百分比；分区表为空
FREELISTS	INTEGER	分配给该段的进程空闲列表数；分区表为空
FREELIST_GROUPS	INTEGER	分配给该段的空闲列表组的数量；分区表为空
LOGGING	VARCHAR2(3)	指示是否记录对表的更改；分区表为空；YES/NO
BACKED_UP	VARCHAR2(1)	指示自最后一次修改(Y)以来是否备份了表(N)
NUM_ROWS	INTEGER	表中的行数
BLOCKS	INTEGER	表中已使用数据块的个数
EMPTY_BLOCKS	INTEGER	表中空(从未使用过)数据块的数量。只有在使用DBMS_STATS包收集表上的统计信息时，才会填充此列
AVG_SPACE	INTEGER	分配给表的数据块中的平均可用空间量(以字节为单位)
CHAIN_CNT	INTEGER	表中从一个数据块链接到另一个数据块或已迁移到新块的行数，需要一个链接来保存旧的ROWID
AVG_ROW_LEN	INTEGER	表中一行的平均长度(以字节为单位)
AVG_SPACE_FREELIST_BLOCKS	INTEGER	自由列表中所有块的平均空闲空间
NUM_FREELIST_BLOCKS	INTEGER	自由列表上的块数
DEGREE	VARCHAR2(10)	每个实例用于扫描表的线程数，或默认值
INSTANCES	VARCHAR2(10)	要扫描表的实例数，或默认值
CACHE	VARCHAR2(5)	指示是否将表缓存在缓冲区缓存(Y)中(N)
TABLE_LOCK	VARCHAR2(8)	表锁定是启用(enabled)还是禁用(disabled)
SAMPLE_SIZE	INTEGER	用于分析表格的样本量
LAST_ANALYZED	DATE	该表最近被分析的日期
PARTITIONED	VARCHAR2(3)	指示表是否分区(YES)或未分区(NO)
IOT_TYPE	VARCHAR2(12)	如果表是索引组织的表，则IOT_TYPE为IOT、IOT_OVERFLOW

名称	类型	描述
		或IOT_MAPPING。如果表不是索引组织的表, 则IOT_TYPE为NULL
TEMPORARY	VARCHAR2(1)	表是临时的(Y)还是非临时的(N)
SECONDARY	VARCHAR2(1)	说明表是否为Oracle数据盒(Y)的ODCIIndexCreate方法创建的辅助对象(N)
NESTED	VARCHAR2(3)	指示表是嵌套表(是)还是非嵌套表(否)
BUFFER_POOL	VARCHAR2(7)	表的缓冲池; 分区表为空
ROW_MOVEMENT	VARCHAR2(8)	指示分区行移动是启用(enabled)还是禁用(disabled)
GLOBAL_STATS	VARCHAR2(3)	如果统计信息被收集或增量维护, GLOBAL_STATS将为YES, 否则为NO
USER_STATS	VARCHAR2(3)	表示统计数据是由用户直接输入的(YES)还是不是(NO)
DURATION	VARCHAR2(15)	临时表的持续时间, 取值如下所示。 <ul style="list-style-type: none"> • SYS\$SESSION - 行在会话期间保留 • SYS\$TRANSACTION - 行在提交后被删除 • Null - 永久表
SKIP_CORRUPT	VARCHAR2(8)	指示Oracle数据库是否在表和索引扫描期间忽略标记为损坏的块(启用)或引发错误(禁用)。要启用此特性, 请运行DBMS_REPAIR.SKIP_CORRUPT_BLOCKS过程
MONITORING	VARCHAR2(3)	死元组
CLUSTER_OWNER	VARCHAR2(63)	表所属集群的所有者(如果有的话)
DEPENDENCIES	VARCHAR2(8)	指示行级依赖项跟踪是启用(启用)还是禁用(禁用)
COMPRESSION	VARCHAR2(8)	表压缩是否启用(启用)或不启用(禁用); 对于分区表, 为NULL
DROPPED	VARCHAR2(3)	指示表是否已被删除, 是否在回收站(YES)或(NO); 对于分区表, 为NULL, 此视图不返回已删除的表的名称

1.136. dba_tablespace

dba_tablespace视图描述数据库中的表空间。

表 1.137. dba_tablespace的列

名称	类型	描述
TABSPACE_NAME	VARCHAR2(63)	命名空间名
INITIAL_EXTENT	NUMERIC	默认的初始区段大小(以字节为单位)
NEXT_EXTENT	NUMERIC	默认的增量区段大小(以字节为单位)
MIN_EXTENTS	NUMERIC	默认的最小区段数
MAX_EXTENTS	NUMERIC	默认的最大区段数
PCT_INCREASE	NUMERIC	区段大小的默认增加百分比
MIN_EXTLEN	NUMERIC	此表空间的最小区段大小(以字节为单位)
STATUS	VARCHAR2(9)	状态，取值如下所示。 <ul style="list-style-type: none"> • ONLINE • OFFLINE • READ ONLY
CONTENTS	VARCHAR2(9)	内容：PERMANENT
LOGGING	VARCHAR2(9)	默认日志属性：
EXTENT_MANAGEMENT	VARCHAR2(10)	指示表空间中的区段是字典管理的(dictionary)还是本地管理的(LOCAL)
ALLOCATION_TYPE	VARCHAR2(9)	表空间有效的区段分配类型
PLUGGED_IN	VARCHAR2(3)	指示表空间是否已插入(YES)或(NO)

1.137. dba_tablespaces

dba_tablespaces视图描述数据库中的所有表空间。

表 1.138. dba_tablespaces的列

名称	类型	描述
TABSPACE_NAME	VARCHAR2(63)	命名空间名
INITIAL_EXTENT	NUMERIC	默认的初始区段大小(以字节为单位)
NEXT_EXTENT	NUMERIC	默认的增量区段大小(以字节为单位)
MIN_EXTENTS	NUMERIC	默认的最小区段数

名称	类型	描述
MAX_EXTENTS	NUMERIC	默认的最大区段数
PCT_INCREASE	NUMERIC	区段大小的默认增加百分比
MIN_EXTLEN	NUMERIC	此表空间的最小区段大小(以字节为单位)
STATUS	VARCHAR2 (9)	状态，取值如下所示。 <ul style="list-style-type: none"> • ONLINE • OFFLINE • READ ONLY
CONTENTS	VARCHAR2 (9)	内容：PERMANENT
LOGGING	VARCHAR2 (9)	默认日志属性
EXTENT_MANAGEMENT	VARCHAR2 (10)	指示表空间中的区段是字典管理的(dictionary)还是本地管理的(LOCAL)
ALLOCATION_TYPE	VARCHAR2 (9)	表空间有效的区段分配类型
PLUGGED_IN	VARCHAR2 (3)	指示表空间是否已插入(YES)或(NO)

1.138. dba_trigger_cols

dba_trigger_cols视图描述数据库中所有触发器的列。

表 1.139. dba_trigger_cols的列

名称	类型	描述
TRIGGER_OWNER	VARCHAR2 (63)	触发器的所有者
TRIGGER_NAME	VARCHAR2 (63)	触发器名
TABLE_OWNER	VARCHAR2 (63)	表的所有者
TABLE_NAME	VARCHAR2 (63)	表名
COLUMN_NAME	VARCHAR2 (4000)	列名

1.139. dba_triggers

dba_triggers视图描述数据库中所有的触发器。

表 1.140. dba_triggers的列

名称	类型	描述
OWNER	VARCHAR2 (63)	对象的所有者
TRIGGER_NAME	VARCHAR2 (63)	触发器名
TRIGGER_TYPE	VARCHAR2 (16)	触发器类型
TRIGGERING_EVENT	VARCHAR2 (216)	触发触发器的DML、DDL或数据库事件

名称	类型	描述
TABLE_OWNER	VARCHAR2 (63)	表的拥有者
TABLE_NAME	VARCHAR2 (63)	表名
BASE_OBJECT_TYPE	VARCHAR2 (16)	定义触发器的基对象，取值如下所示 <ul style="list-style-type: none"> • TABLE • VIEW • SCHEMA • DATABASE
COLUMN_NAME	VARCHAR2 (4000)	列名
REFERENCING_NAMES	VARCHAR2 (422)	用于从触发器中引用OLD和NEW列值的名称
DESCRIPTION	VARCHAR2 (4000)	触发描述；用于重新创建触发器创建语句
STATUS	VARCHAR2 (8)	指示触发器是启用(enabled)还是禁用(disabled)；禁用的触发器不会触发
TRIGGER_BODY	TEXT	触发器触发时执行的语句

1.140. dba_users

dba_users视图描述数据库中所有用户的信息

表 1.141. dba_users的列

名称	类型	描述
USERNAME	VARCHAR2 (63)	用户名
USER_ID	NUMERIC (38, 0)	用户id
PASSWORD	VARCHAR2 (63)	密码
ACCOUNT_STATUS	VARCHAR2 (32)	状态，如下所示 <ul style="list-style-type: none"> • OPEN • EXPIRED
LOCK_DATE	DATE	当用户状态为锁定时，用户被锁定的日期
EXPIRY_DATE	DATE	用户的到期日期
DEFAULT_TABLESPACE	VARCHAR2 (63)	默认数据表空间
TEMPORARY_TABLESPACE	VARCHAR2 (63)	临时表的默认表空间名称或表空间组名称
CREATED	DATE	用户创建的时间
PROFILE	VARCHAR2 (63)	用户资源配置文件名称
INITIAL_RSRC_CONSUMER_GROUP	VARCHAR2 (63)	用户的初始资源消费组

名称	类型	描述
EXTERNAL_NAME	VARCHAR2(4000)	用户外部名称。对于集中管理的用户，如果数据库用户映射是一个独占映射，那么这将是用户的目录服务DN。如果这个数据库用户是一个共享模式，它将是一个组的DN
PASSWORD_VERSIONS	VARCHAR2(8)	版本号 eg: 2. 1. 1. 5B
EDITIONS_ENABLED	VARCHAR2(1)	指示是否为相应的用户 (Y) 启用了版本 (N)

1.141. dba_views

dba_views视图描述当前用户的所有视图的信息。

表 1.142. dba_views的列

名称	类型	描述
OWNER	VARCHAR2(63)	视图的拥有者
VIEW_NAME	VARCHAR2(63)	视图名
TEXT_LENGTH	NUMERIC	视图文本的长度
TEXT	TEXT	查看文本。只有当行起源于当前容器时，此列才返回正确的值。在此视图中，BEQUEATH子句不会作为TEXT列的一部分出现
TYPE_TEXT_LENGTH	NUMERIC	视图的类型子句的长度
TYPE_TEXT	VARCHAR2(4000)	视图的类型子句
OID_TEXT_LENGTH	NUMERIC	视图的WITH OID子句的长度
OID_TEXT	VARCHAR2(4000)	视图的WITH OID子句
VIEW_TYPE_OWNER	VARCHAR2(63)	如果视图是类型化视图，则为视图类型的所有者
VIEW_TYPE	VARCHAR2(63)	如果视图是类型化视图，则为视图的类型
SUPERVIEW_NAME	VARCHAR2(63)	父视图的名称
EDITIONING_VIEW	VARCHAR2(1)	保留供将来使用
READ_ONLY	VARCHAR2(1)	指示视图是否只读 (Y) (N)

1.142. user_arguments

user_arguments视图描述当前用户拥有的过程和函数的参数。

表 1.143. user_arguments的列

名称	类型	描述
PACKAGE_NAME	CHARACTER VARYING	包的名称

名称	类型	描述
OBJECT_NAME	CHARACTER VARYING	过程或函数的名称
OBJECT_ID	OID	对象的对象号
POSITION	INTEGER	此列保存该项在参数列表中的位置，0表示函数返回值
ARGUMENT_NAME	CHARACTER VARYING	参数名称，空参数名用于表示函数返回
IN_OUT	CHARACTER VARYING	入参/出参类型 <ul style="list-style-type: none"> • IN • OUT • IN/OUT
TYPE_NAME	CHARACTER VARYING	实参类型的名称。如果类型是包本地类型(也就是说，它在包规范中声明)，则此列显示包的名称
OVERLOAD	CHARACTER VARYING	指示第n个重载(按其在源中的出现顺序)；否则，它是NULL
SUBPROGRAM_ID	INTEGER	唯一子程序标识符
SEQUENCE	INTEGER	定义参数的顺序。参数序列从1开始。首先是返回类型，然后是每个参数
DATA_LEVEL	INTEGER	复合类型实参的嵌套深度
DATA_TYPE	CHARACTER VARYING	参数的数据类型
DEFAULTED	CHARACTER VARYING	指定参数是否为默认值
DEFAULT_VALUE	TEXT	保留供将来使用
DEFAULT_LENGTH	INTEGER	保留供将来使用
DATA_LENGTH	INTEGER	列长度(以字节为单位)
DATA_PRECISION	INTEGER	十进制数字(NUMBER)或二进制数字(FLOAT)的长度
DATA_SCALE	INTEGER	小数点右边的数字
RADIX	INTEGER	一个数字的基数参数
CHARACTER_SET_NAME	CHARACTER VARYING	参数的字符集名称
TYPE_OWNER	CHARACTER VARYING	参数类型的所有者
TYPE_SUBNAME	CHARACTER VARYING	只与包本地类型相关。显示在TYPE_NAME列标识的包中声明的类型的名称
TYPE_LINK	CHARACTER VARYING	当TYPE_NAME列中标识的包是远程包时，仅与包本地类型相关。此列显示用于引用远程包的数据库链接

名称	类型	描述
TYPE_OBJECT_TYPE	CHARACTER VARYING	显示由TYPE_OWNER、TYPE_NAME和TYPE_SUBNAME列描述的类型。取值如下所示。 <ul style="list-style-type: none"> • TABLE • VIEW • PACKAGE • TYPE
PLS_TYPE	CHARACTER VARYING	对于数值形参，实参的PL/SQL类型的名称。否则无效
CHAR_LENGTH	INTEGER	字符串数据类型的字符限制
CHAR_USED	CHARACTER VARYING	指示字符串的字节限制(B)或字符限制(C)是否正式
ORIGIN_CON_ID	INTEGER	数据产生的容器的ID。可能的值包括： <ul style="list-style-type: none"> • 0: 该值用于非cdb中的行。此值不用于cdb • n: 该值用于包含起源于容器ID为n的容器的数据的行(如果该行起源于root，则n = 1)

1.143. user_col_comments

user_col_comments视图在当前用户拥有的表和视图的列上注释。

表 1.144. user_col_comments的列

名称	类型	描述
OWNER	VARCHAR2(63)	对象的拥有者
TABLE_NAME	VARCHAR2(63)	对象名
COLUMN_NAME	VARCHAR2(63)	列名
COMMENTS	VARCHAR2(4000)	描述

1.144. user_col_privs

user_col_privs视图描述当前用户下的列级权限表。

表 1.145. user_col_privs的列

名称	类型	描述
GRANTOR	VARCHAR2(63)	执行授权的用户的名称
OWNER	VARCHAR2(63)	对象的拥有者

名称	类型	描述
GRANTEE	VARCHAR2 (63)	授予访问权的用户或角色的名称
TABLE_CATALOG	VARCHAR2 (63)	当前的数据库
TABLE_SCHEMA	VARCHAR2 (63)	对象的模式
TABLE_NAME	VARCHAR2 (63)	表名
COLUMN_NAME	VARCHAR2 (63)	列名
PRIVILEGE_TYPE	VARCHAR2 (40)	权限名称
IS_GRANTABLE	VARCHAR2 (3)	允许访问

1.145. user_cons_columns

user_cons_columns视图描述当前用户拥有并在约束中指定的列。

表 1.146. user_cons_columns的列

名称	类型	描述
OWNER	VARCHAR2 (63)	兑现的拥有者
CONSTRAINT_NAME	VARCHAR2 (63)	约束名
TABLE_NAME	VARCHAR2 (63)	表名
COLUMN_NAME	VARCHAR2 (4000)	列名
POSITION	NUMERIC	定义中列的原始位置

1.146. user_constraints

user_constraints视图描述当前用户所拥有的表的所有约束定义。

表 1.147. user_constraints的列

名称	类型	描述
OWNER	VARCHAR2 (63)	对象的拥有者
CONSTRAINT_NAME	VARCHAR2 (63)	约束名
CONSTRAINT_TYPE	VARCHAR2 (1)	约束定义的类型，取值如下所示 <ul style="list-style-type: none"> • C -表上的检查约束 • P -主键 • U -唯一密钥 • R -外键 • T -受限触发器 • X -排他性约束
TABLE_NAME	VARCHAR2 (63)	表名

名称	类型	描述
SEARCH_CONDITION	text	检查约束的搜索条件的文本。只有当行起源于当前容器时，此列才返回正确的值
R_OWNER	VARCHAR2 (63)	引用约束中引用的表的所有者
R_CONSTRAINT_NAME	VARCHAR2 (63)	引用表的唯一约束定义的名称
DELETE_RULE	VARCHAR2 (9)	引用约束的删除规则如下所示。 <ul style="list-style-type: none"> • RESTRICT • CASCADE • NULL • SET NULL • SET DEFAULT • NO ACTION
STATUS	VARCHAR2 (8)	约束的实施状态如下所示。 <ul style="list-style-type: none"> • ENABLED • DISABLED
DEFERRABLE	VARCHAR2 (14)	指示约束是否可延迟 (deferrable) 或不可延迟 (not deferrable)
DEFERRED	VARCHAR2 (9)	指示约束最初是否被延迟 (deferred) (IMMEDIATE)
VALIDATED	VARCHAR2 (13)	当STATUS = ENABLED时，可能的值为： <ul style="list-style-type: none"> • VALIDATED -所有数据都遵守约束 (也就是说，当启用约束时，表中的现有数据以及输入到表中的任何后续数据都将得到验证) • NOVALIDATED -所有数据都可能不遵守约束 (也就是说，当启用约束时，表中现有的数据没有被验证，但是输入到表中的后续数据被验证)
GENERATED	VARCHAR2 (14)	指示约束的名称是用户生成 (USER name) 还是系统生成 (GENERATED name)
BAD	VARCHAR2 (3)	指示此约束是否以模糊的方式指定世纪 (BAD) 或否 (NULL)。为了避免这种模糊性导致的错

名称	类型	描述
		误，可以使用带四位数年份的 TO_DATE 函数重写约束
RELY	VARCHAR2 (4)	当 VALIDATED = NOT VALIDATED 时，此列表示查询重写 (RELY) 时是否考虑约束 (NULL)
LAST_CHANGE	TIMESTAMP (0) WITHOUT TIME ZONE	约束最后一次启用或禁用的时间
INDEX_OWNER	VARCHAR2 (63)	拥有索引的用户名
INDEX_NAME	VARCHAR2 (63)	索引的名称 (仅在唯一和主键约束时显示)
INVALID	VARCHAR2 (7)	指示约束是否无效 (invalid) 或无效 (NULL)
VIEW_RELATED	VARCHAR2 (14)	指示约束是否依赖于视图 (DEPEND on view) (NULL)

1.147. user_ind_columns

user_ind_columns 视图描述数据库中所有建有索引的列的信息。

表 1.148. user_ind_columns 的列

名称	类型	描述
INDEX_NAME	VARCHAR2 (63)	索引名
TABLE_OWNER	VARCHAR2 (63)	表的拥有者
TABLE_NAME	VARCHAR2 (63)	表名
COLUMN_NAME	VARCHAR2 (4000)	列名
COLUMN_POSITION	SMALLINT	索引中的列或属性的位置
COLUMN_LENGTH	NUMERIC	列的索引长度
CHAR_LENGTH	NUMERIC	列的最大码点长度
DESCEND	VARCHAR2 (4)	列是否按降序排序 (Y/N)

1.148. user_indexes

user_indexes 视图描述当前用户拥有的所有索引信息。

表 1.149. user_indexes 的列

名称	类型	描述
INDEX_NAME	VARCHAR2 (63)	索引名
INDEX_TYPE	VARCHAR2 (27)	索引类型
TABLE_OWNER	VARCHAR2 (63)	表的拥有者
TABLE_NAME	VARCHAR2 (63)	表名
TABLE_TYPE	TEXT	表类型
UNIQUENESS	VARCHAR2 (9)	索引的惟一状态

名称	类型	描述
COMPRESSION	VARCHAR2 (8)	用于索引的压缩类型
PREFIX_LENGTH	NUMERIC	压缩键前缀中的列数
TABLESPACE_NAME	VARCHAR2 (63)	包含索引的表空间的名称
INI_TRANS	VARCHAR2 (7)	初始交易数量
MAX_TRANS	VARCHAR2 (7)	最大交易数量
INITIAL_EXTENT	VARCHAR2 (7)	初始范围的大小
NEXT_EXTENT	VARCHAR2 (7)	辅助扩展区的大小
MIN_EXTENTS	VARCHAR2 (7)	段中允许的最小范围数
MAX_EXTENTS	VARCHAR2 (7)	段中允许的最大范围数
PCT_INCREASE	VARCHAR2 (7)	区大小增加的百分比
PCT_THRESHOLD	VARCHAR2 (7)	每个索引条目允许的块空间百分比阈值
INCLUDE_COLUMN	VARCHAR2 (7)	要包含在按索引组织的表主键(非溢出)索引中的最后一列的列ID
FREELISTS	VARCHAR2 (7)	分配给该段的进程空闲列表数
FREELIST_GROUPS	VARCHAR2 (7)	分配给该段的空闲列表组的数量
PCT_FREE	VARCHAR2 (7)	块中可用空间的最小百分比
LOGGING	VARCHAR2 (7)	指示是否记录对索引的更改:
BLEVEL	VARCHAR2 (7)	b *-树级别(从其根块到其叶块的索引深度)。深度0表示根块和叶块是相同的
LEAF_BLOCKS	VARCHAR2 (7)	索引中的叶块数
DISTINCT_KEYS	VARCHAR2 (7)	不同索引值的数量。对于强制执行的索引UNIQUE和PRIMARY KEY约束, 该值与表中的行数相同(*_TABLES.NUM_ROWS)
AVG_LEAF_BLOCKS_PER_KEY	VARCHAR2 (7)	索引中每个不同值出现的叶块数的平均值, 四舍五入为最接近的整数。对于强制执行的索引UNIQUE和PRIMARY KEY约束, 该值始终为1
AVG_DATA_BLOCKS_PER_KEY	VARCHAR2 (7)	表中由索引中的非重复值指向的平均数据块数, 四舍五入为最接近的整数。此统计信息是包含索引列中包含给定值的行的数据块的平均数量
CLUSTERING_FACTOR	VARCHAR2 (7)	根据索引值指示表中行的顺序 如果该值接近块数, 则该表非常有序。在这种情况下, 单个叶块中的索引条目倾向于指向相同数据块中的行

名称	类型	描述
		如果该值接近行数，则该表是随机排序的。在这种情况下，同一叶块中的索引条目不太可能指向同一数据块中的行
STATUS	VARCHAR2 (8)	指示非分区索引是否为VALID或者UNUSABLE
NUM_ROWS	VARCHAR2 (7)	索引中的行数。
SAMPLE_SIZE	VARCHAR2 (7)	用于分析指数的样本大小
LAST_ANALYZED	VARCHAR2 (20)	最近分析此索引的日期
DEGREE	VARCHAR2 (40)	每个实例用于扫描索引的线程数，或者DEFAULT
INSTANCES	VARCHAR2 (40)	要扫描索引的实例数，或者DEFAULT
PARTITIONED	VARCHAR2 (3)	指示索引是否已分区 (YES) 还是不 (NO)
TEMPORARY	VARCHAR2 (1)	指示索引是否在临时表上 (Y) 还是不 (N)
GENERATED	VARCHAR2 (1)	指示索引的名称是否是系统生成的 (Y) 还是不 (N)
SECONDARY	VARCHAR2 (1)	指示该索引是否是由 ODCIIndexCreateOracle数据盒式磁带的方法 (Y) 还是不 (N)
BUFFER_POOL	VARCHAR2 (7)	用于索引块的缓冲池：
USER_STATS	VARCHAR2 (3)	指示统计信息是否由用户直接输入 (YES) 还是不 (NO)
DURATION	VARCHAR2 (15)	指示临时表的持续时间： <ul style="list-style-type: none"> • SYS\$SESSION-在会话期间保留行 • SYS\$TRANSACTION-行在以下时间后被删除COMMIT
PCT_DIRECT_ACCESS	VARCHAR2 (7)	对于索引组织表上的辅助索引，使用 VALID 猜测的行的百分比
ITYP_OWNER	VARCHAR2 (63)	对于域索引，索引类型的所有者
ITYP_NAME	VARCHAR2 (63)	对于域索引，是索引类型的名称
PARAMETERS	VARCHAR2 (1000)	对于域索引，参数字符串
GLOBAL_STATS	VARCHAR2 (3)	如果收集或增量维护统计信息，则GLOBAL_STATS将为YES，否则将为NO
DOMIDX_STATUS	VARCHAR2 (12)	域索引的状态：

名称	类型	描述
		<ul style="list-style-type: none"> 空索引不是域索引 VALID- Index是有效的域索引 IDXTYP_INVLD-域索引的 Indextype无效
DOMIDX_OPSTATUS	VARCHAR2 (6)	域索引的操作状态： <ul style="list-style-type: none"> 空索引不是域索引 VALID-操作执行无误 FAILED-操作失败，出现错误
FUNCIDX_STATUS	VARCHAR2 (8)	基于函数的索引的状态： <ul style="list-style-type: none"> 空索引不是基于函数的索引 ENABLED-启用基于函数的索引 DISABLED-基于函数的索引被禁用
JOIN_INDEX	TEXT	指示该索引是否是联接索引 (YES) 还是不 (NO)
IOT_REDUNDANT_PKEY_ELIM	TEXT	指示是否从按索引组织的表上的辅助索引中删除了冗余的主键列 (YES) 还是不 (NO)
DROPPED	TEXT	指示该索引是否已被删除并在回收站中 (YES) 还是不 (NO)；分区表为空。该视图不返回已被删除的索引的名称

1. 149. user_objects

user_objects视图描述当前用户拥有的所有对象。

表 1. 150. user_objects的列

名称	类型	描述
OBJECT_NAME	VARCHAR2 (63)	对象名
SUBOBJECT_NAME	VARCHAR2 (63)	子对象的名称
OBJECT_ID	NUMERIC (38, 0)	对象的对象号
DATA_OBJECT_ID	NUMERIC (38, 0)	包含该对象的段的字典对象号
OBJECT_TYPE	VARCHAR2 (19)	对象类型
CREATED	DATE	对象创建的时间戳
LAST_DDL_TIME	DATE	对象和DDL语句产生的依赖对象的最后修改的时间戳(包括授予和撤销)

名称	类型	描述
TIMESTAMP	VARCHAR2 (20)	对象规范的时间戳(字符数据)
STATUS	VARCHAR2 (7)	对象的状态
TEMPORARY	VARCHAR2 (1)	对象是否是临时的
GENERATED	VARCHAR2 (1)	是否生成此对象系统的名称
SECONDARY	VARCHAR2 (1)	指示这是否是Oracle Data Cartridge的ODCIIndexCreate方法创建的辅助对象
NAMESPACE	NUMERIC (38, 0)	命名空间

1.150. user_role_privs

user_role_privs视图描述了授予当前用户的角色。

表 1.151. user_role_privs的列

名称	类型	描述
USERNAME	VARCHAR2 (63)	接收授权的用户或角色的名称
GRANTED_ROLE	VARCHAR2 (63)	授予的角色名
ADMIN_OPTION	VARCHAR2 (3)	指示授予是否带有管理选项 (YES) 或否 (NO)
DEFAULT_ROLE	VARCHAR2 (3)	指示角色是否被指定为用户的默认角色 (YES) 或否 (NO)

1.151. user_sequences

user_sequences视图描述当前用户的所有序列的信息。

表 1.152. user_sequences的列

名称	类型	描述
SEQUENCE_OWNER	VARCHAR2 (63)	序列的拥有者
SEQUENCE_NAME	VARCHAR2 (63)	序列名
MIN_VALUE	NUMERIC (38, 0)	序列的最小值
MAX_VALUE	NUMERIC (38, 0)	序列的最大值
INCREMENT_BY	NUMERIC (38, 0)	序列按该值递增
CYCLE_FLAG	TEXT	指示序列在达到极限 (Y) 时是否换行 (N)
ORDER_FLAG	VARCHAR2 (1)	指示序列号是否按 (Y) 的顺序生成 (N)
CACHE_SIZE	NUMERIC (38, 0)	要缓存的序列号的个数
LAST_NUMERIC	NUMERIC (38, 0)	最后写入磁盘的序列号。如果序列使用缓存, 则写入磁盘的数字是序列缓存中的最后一个数字。这个数字很可能大于所

名称	类型	描述
		使用的最后一个序列号。对于会话序列，应忽略此列中的值

1.152. user_source

user_source视图描述当前用户的所有程序源的信息。

表 1.153. user_source的列

名称	类型	描述
OWNER	VARCHAR2 (63)	对象的拥有者
NAME	VARCHAR2 (63)	对象名
TYPE	VARCHAR2 (12)	对象类型:: FUNCTION, JAVA SOURCE, PACKAGE, PACKAGE BODY, PROCEDURE, TRIGGER, TYPE, TYPE BODY
TEXT	VARCHAR2 (4000)	存储对象的文本源

1.153. user_synonyms

user_synonyms视图描述数据库中用户所有同义词的信息。

表 1.154. user_synonyms的列

名称	类型	描述
OWNER	VARCHAR2 (63)	同义词的所有者
SYNONYM_NAMESPACE_NAME	VARCHAR2 (63)	同义词的命名空间
SYNONYM_NAME	VARCHAR2 (63)	同义词名
TABLE_OWNER	VARCHAR2 (63)	由同义词引用的对象的所有者
TABLE_NAMESPACE_NAME	VARCHAR2 (63)	表的命名空间
TABLE_NAME	VARCHAR2 (63)	由同义词引用的对象的名称
DBLINK	TEXT	引用的数据库链接的名称(如果有的话)

1.154. user_tab_cols

user_tab_cols视图描述当前用户可以访问的表，视图的列。

表 1.155. user_tab_cols的列

名称	类型	描述
TABLE_NAME	VARCHAR2 (63)	表、视图的名称
COLUMN_NAME	VARCHAR2 (63)	列名
DATA_TYPE	VARCHAR2 (106)	列的数据类型
DATA_TYPE_MOD	VARCHAR2 (3)	列的数据类型修饰符

名称	类型	描述
DATA_TYPE_OWNER	VARCHAR2(63)	列的数据类型的所有者
DATA_LENGTH	NUMERIC	列长度(以字节为单位)
DATA_PRECISION	NUMERIC	数字数据类型的十进制精度;浮点型的二进制精度;所有其他数据类型为空
DATA_SCALE	NUMERIC	位数:数字中小数点右边的位数
NULLABLE	VARCHAR2(1)	指示列是否允许为空。如果列上存在非NULL约束或列是主键的一部分,则该值为N
COLUMN_ID	NUMERIC	创建的列的序列号
DEFAULT_LENGTH	NUMERIC	列的默认值的长度
DATA_DEFAULT	TEXT	列的默认值
NUM_DISTINCT	NUMERIC	列中不同值的数目
LOW_VALUE	NUMERIC	列中的低值
HIGH_VALUE	NUMERIC	列中的高值
DENSITY	NUMERIC	如果柱状图在COLUMN_NAME上可用,则此列显示在柱状图中跨越少于2个端点的值的选择性。它不表示跨越2个或多个端点的值的选择性。如果柱状图在COLUMN_NAME上不可用,则该列的值为1/NUM_DISTINCT
NUM_NULLS	NUMERIC	列中空值的数目
NUM_BUCKETS	NUMERIC	列的直方图中的桶数。注意:直方图中的桶数是在ANALYZE SQL语句的SIZE参数中指定的。然而,Oracle数据库并没有创建一个比样本行数更多的桶的直方图。此外,如果样本中包含任何重复的值,Oracle数据库会创建指定数量的桶,但由于内部压缩算法的原因,这一列显示的值可能更小
LAST_ANALYZED	TIMESTAMP(0) WITHOUT TIME ZONE	这一栏最近分析的日期
SAMPLE_SIZE	NUMERIC	用于分析该列的样本量
CHARACTER_SET_NAME	VARCHAR2(44)	字符集名称:CHAR_CS/NCHAR_CS
CHAR_COL_DECL_LENGTH	NUMERIC	字符类型列的声明长度
GLOBAL_STATS	VARCHAR2(3)	如果统计信息被收集或增量维护,GLOBAL_STATS将为YES,否则为NO
USER_STATS	VARCHAR2(3)	指示统计信息是否由用户直接输入(YES)或(NO)

名称	类型	描述
AVG_COL_LEN	NUMERIC	列的平均长度(以字节计)
CHAR_LENGTH	NUMERIC	以字符为单位显示列的长度。 该值仅适用于以下数据类型。 <ul style="list-style-type: none"> • CHAR • VARCHAR2 • NCHAR • NVARCHAR2
CHAR_USED	VARCHAR2(1)	指示列使用BYTE长度语义(B)或CHAR长度语义(C), 或者数据类型不是以下任何一种(NULL)。 <ul style="list-style-type: none"> • CHAR • VARCHAR2 • NCHAR • NVARCHAR2
V80_FMT_IMAGE	VARCHAR2(3)	指示列数据是否为8.0版本图像格式(YES)或否(NO)
DATA_UPGRADED	VARCHAR2(3)	指示列数据是否已升级为最新类型版本格式(YES)或(NO)
HIDDEN_COLUMN	VARCHAR2(3)	指示列是否是隐藏列(YES)或不是(NO)
VIRTUAL_COLUMN	VARCHAR2(3)	指示列是否是虚拟列(YES)或不是(NO)
SEGMENT_COLUMN_ID	NUMERIC	段中列的序列号
INTERNAL_COLUMN_ID	NUMERIC	列的内部序列号
HISTOGRAM	VARCHAR2(15)	表示直方图的存在/类型。取值如下所示 <ul style="list-style-type: none"> • NONE • FREQUENCY • TOP-FREQUENCY • HEIGHT BALANCED • HYBRID
QUALIFIED_COL_NAME	VARCHAR2(4000)	限定的列名

1.155. user_tab_columns

user_tab_columns视图描述数据库中所有表列的信息。

表 1.156. user_tab_columns的列

名称	类型	描述
TABLE_NAME	VARCHAR2(63)	表、视图的名称
COLUMN_NAME	VARCHAR2(63)	列名
DATA_TYPE	VARCHAR2(106)	列的数据类型
DATA_TYPE_MOD	VARCHAR2(3)	列的数据类型修饰符
DATA_TYPE_OWNER	VARCHAR2(63)	列的数据类型的所有者
DATA_LENGTH	NUMERIC	列长度(以字节为单位)
DATA_PRECISION	NUMERIC	数字数据类型的十进制精度;浮点型的二进制精度;所有其他数据类型为空
DATA_SCALE	NUMERIC	位数:数字中小数点右边的位数
NULLABLE	VARCHAR2(1)	指示列是否允许为空。如果列上存在非NULL约束或列是主键的一部分,则该值为N
COLUMN_ID	NUMERIC	创建的列的序列号
DEFAULT_LENGTH	NUMERIC	列的默认值的长度
DATA_DEFAULT	TEXT	列的默认值
NUM_DISTINCT	NUMERIC	列中不同值的数目
LOW_VALUE	NUMERIC	列中的低值
HIGH_VALUE	NUMERIC	列中的高值
DENSITY	NUMERIC	如果柱状图在COLUMN_NAME上可用,则此列显示在柱状图中跨越少于2个端点的值的选择性。它不表示跨越2个或多个端点的值的选择性。如果柱状图在COLUMN_NAME上不可用,则该列的值为1/NUM_DISTINCT
NUM_NULLS	NUMERIC	列中空值的数目
NUM_BUCKETS	NUMERIC	列的直方图中的桶数。注意:直方图中的桶数是在ANALYZE SQL语句的SIZE参数中指定的。然而,Oracle数据库并没有创建一个比样本行数更多的桶的直方图。此外,如果样本中包含任何重复的值,Oracle数据库会创建指定数量的桶,但由于内部压缩算法的原因,这一列显示的值可能更小
LAST_ANALYZED	TIMESTAMP(0) WITHOUT TIME ZONE	这一栏最近分析的日期
SAMPLE_SIZE	NUMERIC	用于分析该列的样本量
CHARACTER_SET_NAME	VARCHAR2(44)	字符集名称:CHAR_CS/NCHAR_CS
CHAR_COL_DECL_LENGTH	NUMERIC	字符类型列的声明长度

名称	类型	描述
GLOBAL_STATS	VARCHAR2 (3)	如果统计信息被收集或增量维护，GLOBAL_STATS将为YES，否则为NO
USER_STATS	VARCHAR2 (3)	指示统计信息是否由用户直接输入 (YES) 或 (NO)
AVG_COL_LEN	NUMERIC	列的平均长度 (以字节计)
CHAR_LENGTH	NUMERIC	以字符为单位显示列的长度。该值仅适用于以下数据类型。 <ul style="list-style-type: none"> • CHAR • VARCHAR2 • NCHAR • NVARCHAR2
CHAR_USED	VARCHAR2 (1)	指示列使用BYTE长度语义 (B) 或 CHAR长度语义 (C)，或者数据类型不是以下任何一种 (NULL)。 <ul style="list-style-type: none"> • CHAR • VARCHAR2 • NCHAR • NVARCHAR2
V80_FMT_IMAGE	VARCHAR2 (3)	指示列数据是否为8.0版本图像格式 (YES) 或否 (NO)
DATA_UPGRADED	VARCHAR2 (3)	指示列数据是否已升级为最新类型版本格式 (YES) 或 (NO)
HISTOGRAM	VARCHAR2 (15)	表示直方图的存在/类型。取值如下所示。 <ul style="list-style-type: none"> • NONE • FREQUENCY • TOP-FREQUENCY • HEIGHT BALANCED • HYBRID

1.156. user_tab_comments

user_tab_comments视图描述当前用户拥有的表和视图上的注释。

表 1.157. user_tab_comments的列

名称	类型	描述
OWNER	VARCHAR2 (63)	对象的拥有者
TABLE_NAME	VARCHAR2 (63)	表名
TABLE_TYPE	VARCHAR2 (11)	表类型
COMMENTS	VARCHAR2 (4000)	注释

1.157. user_tab_privs

user_tab_privs视图描述当前用户的对象权限，该用户可以是对象的所有者、授予者或者被授予者。

表 1.158. user_tab_privs的列

名称	类型	描述
GRANTEE	VARCHAR2 (63)	授予访问权的用户或角色的名称
OWNER	VARCHAR2 (63)	对象的所有者
TABLE_NAME	VARCHAR2 (63)	对象名称
GRANTOR	VARCHAR2 (63)	执行授权的用户的名称
PRIVILEGE	VARCHAR2 (40)	对象上的特权
GRANTABLE	VARCHAR2 (3)	指示是否使用GRANT OPTION (YES) 或否 (NO) 授予特权

1.158. user_tables

user_tables视图描述当前用户的所有表的信息。

表 1.159. user_tables的列

名称	类型	描述
TABLE_NAME	VARCHAR2 (63)	表的名称
TABLESPACE_NAME	VARCHAR2 (63)	包含该表的表空间的名称；对于分区表、临时表和按索引组织的表，为NULL
CLUSTER_NAME	VARCHAR2 (63)	表所属的集群的名称(如果有)
IOT_NAME	VARCHAR2 (63)	溢出或映射表条目所属的按索引组织的表(如果有)的名称。如果IOT_TYPE列不为空，则此列包含基表名
STATUS	VARCHAR2 (63)	如果以前的DROP TABLE操作失败，指示表是否不可用 (UNUSABLE) 或有效 (VALID)
PCT_FREE	INTEGER	块中可用空间的最小百分比；分区表为空

名称	类型	描述
PCT_USED	INTEGER	块中已用空间的最小百分比；分区表为空
INI_TRANS	INTEGER	处理的初始数量；分区表为空
MAX_TRANS	INTEGER	最大处理数量；分区表为空
INITIAL_EXTENT	INTEGER	初始范围的大小(以字节为单位)；分区表为空
NEXT_EXTENT	INTEGER	二级区段的大小(以字节为单位)；NULL用于分区表
MIN_EXTENTS	INTEGER	段中允许的最小范围数；分区表为空
MAX_EXTENTS	INTEGER	段中允许的最大范围数；分区表为空
PCT_INCREASE	INTEGER	范围大小的增加百分比；分区表为空
FREELISTS	INTEGER	分配给该段的进程空闲列表数；分区表为空
FREELIST_GROUPS	INTEGER	分配给该段的空闲列表组的数量；分区表为空
LOGGING	VARCHAR2(3)	指示是否记录对表的更改；分区表为空；YES/NO
BACKED_UP	VARCHAR2(1)	指示自最后一次修改(Y)以来是否备份了表(N)
NUM_ROWS	INTEGER	表中的行数
BLOCKS	INTEGER	表中已使用数据块的个数
EMPTY_BLOCKS	INTEGER	表中空(从未使用过)数据块的数量。只有在使用DBMS_STATS包收集表上的统计信息时，才会填充此列
AVG_SPACE	INTEGER	分配给表的数据块中的平均可用空间量(以字节为单位)
CHAIN_CNT	INTEGER	表中从一个数据块链接到另一个数据块或已迁移到新块的行数，需要一个链接来保存旧的ROWID
AVG_ROW_LEN	INTEGER	表中一行的平均长度(以字节为单位)
AVG_SPACE_FREELIST_BLOCKS	INTEGER	自由列表中所有块的平均空闲空间
NUM_FREELIST_BLOCKS	INTEGER	自由列表上的块数
DEGREE	VARCHAR2(10)	每个实例用于扫描表的线程数，或默认值
INSTANCES	VARCHAR2(10)	要扫描表的实例数，或默认值
CACHE	VARCHAR2(5)	指示是否将表缓存在缓冲区缓存(Y)中(N)

名称	类型	描述
TABLE_LOCK	VARCHAR2 (8)	表锁定是启用(enabled)还是禁用(disabled)
SAMPLE_SIZE	INTEGER	用于分析表格的样本量
LAST_ANALYZED	DATE	该表最近被分析的日期
PARTITIONED	VARCHAR2 (3)	指示表是否分区(YES)或未分区(NO)
IOT_TYPE	VARCHAR2 (12)	如果表是索引组织的表, 则 IOT_TYPE为IOT、IOT_OVERFLOW或IOT_MAPPING。如果表不是索引组织的表, 则IOT_TYPE为NULL
TEMPORARY	VARCHAR2 (1)	表是临时的(Y)还是非临时的(N)
SECONDARY	VARCHAR2 (1)	说明表是否为Oracle数据盒(Y)的ODCIIndexCreate方法创建的辅助对象(N)
NESTED	VARCHAR2 (3)	指示表是嵌套表(是)还是非嵌套表(否)
BUFFER_POOL	VARCHAR2 (7)	表的缓冲池; 分区表为空
ROW_MOVEMENT	VARCHAR2 (8)	指示分区行移动是启用(enabled)还是禁用(disabled)
GLOBAL_STATS	VARCHAR2 (3)	如果统计信息被收集或增量维护, GLOBAL_STATS将为YES, 否则为NO
USER_STATS	VARCHAR2 (3)	表示统计数据是由用户直接输入的(YES)还是不是(NO)
DURATION	VARCHAR2 (15)	临时表的持续时间, 取值如下所示。 <ul style="list-style-type: none"> • SYS\$SESSION - 行在会话期间保留 • SYS\$TRANSACTION - 行在提交后被删除 • Null - 永久表
SKIP_CORRUPT	VARCHAR2 (8)	指示Oracle数据库是否在表和索引扫描期间忽略标记为损坏的块(启用)或引发错误(禁用)。要启用此特性, 请运行DBMS_REPAIR.SKIP_CORRUPT_BLOCKS过程
MONITORING	VARCHAR2 (3)	死元组
CLUSTER_OWNER	VARCHAR2 (63)	表所属集群的所有者(如果有的话)

名称	类型	描述
DEPENDENCIES	VARCHAR2 (8)	指示行级依赖项跟踪是启用(启用)还是禁用(禁用)
COMPRESSION	VARCHAR2 (8)	表压缩是否启用(启用)或不启用(禁用);对于分区表, 为NULL
DROPPED	VARCHAR2 (3)	指示表是否已被删除, 是否在回收站(YES)或(NO); 对于分区表, 为NULL, 此视图不返回已删除的表的名称

1.159. user_tablespace

user_tablespace视图描述当前用户可访问的表空间。

表 1.160. user_tablespace的列

名称	类型	描述
TABLESPACE_NAME	VARCHAR2 (63)	命名空间名
INITIAL_EXTENT	NUMERIC	默认的初始区段大小(以字节为单位)
NEXT_EXTENT	NUMERIC	默认的增量区段大小(以字节为单位)
MIN_EXTENTS	NUMERIC	默认的最小区段数
MAX_EXTENTS	NUMERIC	默认的最大区段数
PCT_INCREASE	NUMERIC	区段大小的默认增加百分比
MIN_EXTLEN	NUMERIC	此表空间的最小区段大小(以字节为单位)
STATUS	VARCHAR2 (9)	状态, 取值如下所示。 <ul style="list-style-type: none"> • ONLINE • OFFLINE • READ ONLY
CONTENTS	VARCHAR2 (9)	内容: PERMANENT
LOGGING	VARCHAR2 (9)	默认日志属性:
EXTENT_MANAGEMENT	VARCHAR2 (10)	指示表空间中的区段是字典管理的(dictionary)还是本地管理的(LOCAL)
ALLOCATION_TYPE	VARCHAR2 (9)	表空间有效的区段分配类型

1.160. user_tablespaces

user_tablespaces视图描述当前用户可访问的表空间。

表 1.161. user_tablespaces的列

名称	类型	描述
TABSPACE_NAME	VARCHAR2 (63)	命名空间名
INITIAL_EXTENT	NUMERIC	默认的初始区段大小(以字节为单位)
NEXT_EXTENT	NUMERIC	默认的增量区段大小(以字节为单位)
MIN_EXTENTS	NUMERIC	默认的最小区段数
MAX_EXTENTS	NUMERIC	默认的最大区段数
PCT_INCREASE	NUMERIC	区段大小的默认增加百分比
MIN_EXTLEN	NUMERIC	此表空间的最小区段大小(以字节为单位)
STATUS	VARCHAR2 (9)	状态，取值如下所示。 <ul style="list-style-type: none"> • ONLINE • OFFLINE • READ ONLY
CONTENTS	VARCHAR2 (9)	内容：PERMANENT
LOGGING	VARCHAR2 (9)	默认日志属性
EXTENT_MANAGEMENT	VARCHAR2 (10)	指示表空间中的区段是字典管理的(dictionary)还是本地管理的(LOCAL)
ALLOCATION_TYPE	VARCHAR2 (9)	表空间有效的区段分配类型

1.161. user_trigger_cols

user_trigger_cols视图描述当前用户用的所有触发器的列信息。

表 1.162. user_trigger_cols的列

名称	类型	描述
TRIGGER_OWNER	VARCHAR2 (63)	触发器的拥有者
TRIGGER_NAME	VARCHAR2 (63)	触发器名
TABLE_OWNER	VARCHAR2 (63)	表的拥有者
TABLE_NAME	VARCHAR2 (63)	表名
COLUMN_NAME	VARCHAR2 (4000)	列名

1.162. user_triggers

user_triggers视图描述当前用户的所有触发器的信息。

表 1.163. user_triggers的列

名称	类型	描述
TRIGGER_NAME	VARCHAR2 (63)	触发器名
TRIGGER_TYPE	VARCHAR2 (16)	触发器类型
TRIGGERING_EVENT	VARCHAR2 (216)	触发触发器的DML、DDL或数据库事件
TABLE_OWNER	VARCHAR2 (63)	表的拥有者
TABLE_NAME	VARCHAR2 (63)	表名
BASE_OBJECT_TYPE	VARCHAR2 (16)	定义触发器的基对象，取值如下所示 <ul style="list-style-type: none"> • TABLE • VIEW • SCHEMA • DATABASE
COLUMN_NAME	VARCHAR2 (4000)	列名
REFERENCING_NAMES	VARCHAR2 (422)	用于从触发器中引用OLD和NEW列值的名称
DESCRIPTION	VARCHAR2 (4000)	触发描述；用于重新创建触发器创建语句
STATUS	VARCHAR2 (8)	指示触发器是启用(enabled)还是禁用(disabled)；禁用的触发器不会触发
TRIGGER_BODY	TEXT	触发器触发时执行的语句

1.163. user_users

user_users视图描述当前用户。

表 1.164. user_users的列

名称	类型	描述
USERNAME	VARCHAR2 (63)	用户名
USER_ID	NUMERIC (38, 0)	用户id
ACCOUNT_STATUS	VARCHAR2 (32)	状态，取值如下所示 <ul style="list-style-type: none"> • OPEN • EXPIRED
LOCK_DATE	DATE	当用户状态为锁定时，用户被锁定的日期
EXPIRY_DATE	DATE	用户的到期日期
DEFAULT_TABLESPACE	VARCHAR2 (63)	默认数据表空间

名称	类型	描述
TEMPORARY_TABLESPACE	VARCHAR2 (63)	临时表的默认表空间名称或表空间组名称
CREATED	DATE	用户创建的时间
INITIAL_RSRC_CONSUMER_GROUP	VARCHAR2 (63)	用户的初始资源消费组
EXTERNAL_NAME	VARCHAR2 (63)	用户外部名称。对于集中管理的用户，如果数据库用户映射是一个独占映射，那么这将是用户的目录服务DN。如果这个数据库用户是一个共享模式，它将是一个组的DN

1.164. user_views

user_views视图描述当前用户的所有视图的信息。

表 1.165. user_views的列

名称	类型	描述
VIEW_NAME	VARCHAR2 (63)	视图名
TEXT_LENGTH	NUMERIC	视图文本的长度
TEXT	TEXT	查看文本。只有当行起源于当前容器时，此列才返回正确的值。在此视图中，BEQUEATH子句不会作为TEXT列的一部分出现
TYPE_TEXT_LENGTH	NUMERIC	视图的类型子句的长度
TYPE_TEXT	VARCHAR2 (4000)	视图的类型子句
OID_TEXT_LENGTH	NUMERIC	视图的WITH OID子句的长度
OID_TEXT	VARCHAR2 (4000)	视图的WITH OID子句
VIEW_TYPE_OWNER	VARCHAR2 (63)	如果视图是类型化视图，则为视图类型的所有者
VIEW_TYPE	VARCHAR2 (63)	如果视图是类型化视图，则为视图的类型
SUPERVIEW_NAME	VARCHAR2 (63)	父视图的名称
EDITIONING_VIEW	VARCHAR2 (1)	保留供将来使用
READ_ONLY	VARCHAR2 (1)	指示视图是否只读(Y) (N)

1.165. V\$DATABASE

V\$DATABASE视图描述当前数据库信息。

表 1.166. V\$DATABASE的列

名称	类型	描述
DBID	NUMERIC (38, 0)	在创建数据库并存储在所有文件头中时计算的数据库标识符
NAME	VARCHAR2 (63)	数据库名
CREATED	DATE	数据库的创建日期。如果使用 CREATE CONTROLFILE语句重新创建了控件文件，那么此列将显示重新创建控件文件的日期
LOG_MODE	VARCHAR2 (12)	存档日志模式，取值如下所示 <ul style="list-style-type: none"> • NOARCHIVELOG • ARCHIVELOG
CHECKPOINT_CHANGE	NUMERIC	最后一个SCN检查点
CONTROLFILE_TIME	DATE	备份控制文件中的最后时间戳；如果控制文件不是备份，则为空
OPEN_MODE	VARCHAR2 (10)	打开模式信息：READ WRITE
ARCHIVELOG_CHANGE	NUMERIC	存档日志的最高 NEXT_CHANGE# (来自V\$ARCHIVED_LOG视图)
CURRENT_SCN	NUMERIC	当前的视交叉上核；如果数据库当前未打开，则为空。对于一个备用数据库，它是在媒体恢复期间挂载的物理备用数据库的检查点SCN，总是小于V\$RECOVERY_PROGRESS中跟踪的最后一个应用SCN
MAX_SIZE	NUMERIC (38, 0)	最大值
TOTAL_SIZE	NUMERIC (38, 0)	总值
STATUS\$	VARCHAR2 (10)	状态

1.166. V\$INSTANCE

V\$INSTANCE视图描述当前实例信息。

表 1.167. V\$INSTANCE的列

名称	类型	描述
INSTANCE_NAME	VARCHAR2 (16)	用于实例注册的实例号 (对应于 INSTANCE_NUMBER初始化参数)
HOST_NAME	VARCHAR2 (64)	主机的名称
VERSION	VARCHAR2 (17)	数据库版本号
STARTUP_TIME	DATE	实例启动的时间
STATUS	VARCHAR2 (12)	实例的STARTED状态

名称	类型	描述
ARCHIVER	VARCHAR2 (7)	自动归档状态 “STOPPED”
LOGIN	VARCHAR2 (10)	指示实例是否处于无限制模式 (允许所有用户登录) (允许, 还是处于受限模式 (只允许数据库管理员登录) (restricted)
SHUTDOWN_PENDING	VARCHAR2 (3)	指示关闭是否挂起 (YES) 或 (NO)
DATABASE_STATUS	VARCHAR2 (17)	数据库的 “ONLINE” 状态
INSTANCE_ROLE	VARCHAR2 (18)	指示实例是活动的实例 (PRIMARY_INSTANCE) 还是不活动的辅助实例 (SECONDARY_INSTANCE), 如果实例已启动但未挂载, 则指示该实例是未知的
ACTIVE_STATE	VARCHAR2 (9)	实例的静默状态: NORMAL
BLOCKED	VARCHAR2 (3)	指示是否阻塞所有服务 (YES) 或 (NO)

1. 167. V\$LOCK

V\$LOCK视图列出数据库当前持有的锁以及对锁的未完成请求。

表 1. 168. V\$LOCK的列

名称	类型	描述
ADDR	VARCHAR2 (16)	锁状态对象的地址
KADDR	VARCHAR2 (16)	锁的地址
SID	NUMERIC	会话持有或获取锁的标识符
TYPE	VARCHAR2 (2)	用户或系统锁的类型, TM/TX/UL等
LMODE	NUMERIC	会话持有锁的锁模式, 取值如下所示 <ul style="list-style-type: none"> • 0 - none • 1 - null (NULL) • 2 - row-S (SS) • 3 - row-X (SX) • 4 - share (S) • 5 - S/Row-X (SSX) • 6 - exclusive (X)
REQUEST	NUMERIC	进程请求锁的锁模式, 取值如下所示 <ul style="list-style-type: none"> • 0 - none

名称	类型	描述
		<ul style="list-style-type: none"> • 1 - null (NULL) • 2 - row-S (SS) • 3 - row-X (SX) • 4 - share (S) • 5 - S/Row-X (SSX) • 6 - exclusive (X)
CTIME	NUMERIC	自授予当前模式以来的时间
BLOCK	NUMERIC	<p>指示有问题的锁是否正在阻塞其他进程。取值如下所示</p> <ul style="list-style-type: none"> • 0 - 锁没有阻塞任何其他进程 • 1 - 锁正在阻塞其他进程 • 2 - 锁没有阻塞本地节点上任何被阻塞的进程，但可能会阻塞远程节点上的进程，也可能不会阻塞。该值只在 Oracle RAC (Real Application Clusters, Oracle RAC) 配置中使用(不用于单实例配置)

1. 168. V\$LOCKED_OBJECT

V\$LOCKED_OBJECT视图描述当前被锁定的对象。

表 1. 169. V\$LOCKED_OBJECT的列

名称	类型	描述
XIDUSN	NUMERIC	Undo号
XIDSLOT	NUMERIC	槽数
XIDSQN	NUMERIC	序列号
OBJECT_ID	NUMERIC	对象ID被锁定
SESSION_ID	NUMERIC	会话ID
ORACLE_USERNAME	VARCHAR2(63)	用户名
OS_USER_NAME	VARCHAR2(63)	操作系统用户名
PROCESS	VARCHAR2(63)	操作系统进程ID
LOCKED_MODE	NUMERIC	<p>锁定模式。这一列的数值映射到表锁的锁定模式的文本值：</p> <ul style="list-style-type: none"> • 0 - NONE: 请求但尚未获得锁 • 1 - 空

名称	类型	描述
		<ul style="list-style-type: none"> • 2 - ROWS_S (SS):行共享锁 • 3 - ROW_X (SX):行独占表锁 • 4 - SHARE (S):共享表锁定 • 5 - S/ Row - x (SSX):共享行独占表锁 • 6 - 排他(X):排他表锁

1. 169. V\$PARAMETER

V\$PARAMETER视图列出影响当前会话的初始化参数信息。

表 1. 170. V\$PARAMETER的列

名称	类型	描述
NUM	NUMERIC	参数数量
NAME	VARCHAR2 (80)	参数名
TYPE	NUMERIC	参数类型，如下所示 <ul style="list-style-type: none"> • 1 - Boolean • 2 - String • 3 - Integer • 4 - Parameter file • 5 - Reserved • 6 - Big integer
VALUE	VARCHAR2 (512)	会话的参数值(如果在会话中修改);否则为实例范围的参数值
DISPLAY_VALUE	VARCHAR2 (512)	用户友好的格式。例如，如果VALUE列显示一个大整数参数的值262144，那么DISPLAY_VALUE列将显示值256K
ISSES_MODIFIABLE	VARCHAR2 (5)	指示是否可以使用ALTER SESSION (TRUE)更改参数(FALSE)
ISPG_MODIFIALBE	VARCHAR2 (9)	指示是否可以在PDB内部修改参数(TRUE) (FALSE)
ISMODIFIED	VARCHAR2 (10)	指示该参数在实例启动后是否被修改 <ul style="list-style-type: none"> • MODIFIED -参数已被ALTER SESSION修改

名称	类型	描述
		<ul style="list-style-type: none"> • SYSTEM_MOD—使用ALTER SYSTEM修改了参数(这会导致当前登录的所有会话的值都被修改) • FALSE -实例启动后参数未被修改
DESCRIPTION	VARCHAR2 (255)	参数说明

1. 170. V\$SESSION

V\$SESSION视图描述当前会话信息。

表 1. 171. V\$SESSION的列

名称	类型	描述
SADDR	VARCHAR2 (20)	会话地址
SID	NUMERIC	会话标识符
SERIAL#	NUMERIC	会话序列号。用于唯一标识会话的对象。确保在会话结束而另一个会话开始时使用相同的会话ID时，将会话级别的命令应用到正确的会话对象
AUDSID	NUMERIC	审计会话ID 如果数据库配置为统一审计，则此列显示统一审计会话ID 如果将数据库配置为混合模式审计，那么此列将显示传统的审计会话ID
PADDR	VARCHAR2 (20)	拥有会话的进程的地址
USER#	NUMERIC (38, 0)	用户标识符
USERNAME	VARCHAR2 (63)	用户名
COMMAND	NUMERIC	命令正在执行(最后一条语句已解析)。该COMMAND列的值为0表示该命令没有记录在V\$SESSION中
OWNERID	NUMERIC	拥有可迁移会话的用户标识符；如果该值为2147483644，则列内容无效；对于使用Parallel slave的操作，将此值解释为一个4字节的值。低阶2字节表示会话号，高阶字节表示查询协调器的实例ID
TADDR	VARCHAR2 (20)	事务状态对象的地址
LOCKWAIT	VARCHAR2 (8)	会话正在等待的锁的地址；如果没有则为NULL
STATUS	VARCHAR2 (8)	会话状态，取值如下所示

名称	类型	描述
		<ul style="list-style-type: none"> • ACTIVE - 当前执行SQL的会话 • INACTIVE — 未激活的会话，没有配置的限制或尚未超过配置的限制 • KILLED - 被标记为被终止的会话 • CACHED - Oracle*XA使用的临时缓存会话 • SNIPED - 一个已超过某些配置限制(例如，为资源管理器消费组指定的资源限制或在用户配置文件中指定的 idle_time)的非活动会话。这样的会话将不被允许再次激活
SERVER	VARCHAR2 (9)	服务器类型，取值如下所示 <ul style="list-style-type: none"> • DEDICATED • SHARED • PSEUDO • POOLED • NONE
SCHEMA#	NUMERIC (38, 0)	模式用户标识符
SCHEMANAME	VARCHAR2 (63)	模式用户名
OSUSER	VARCHAR2 (63)	操作系统客户端用户名
PROCESS	VARCHAR2 (12)	操作系统客户端进程ID
MACHINE	VARCHAR2 (64)	操作系统机器名称
TERMINAL	VARCHAR2 (63)	操作系统终端名称
PROGRAM	VARCHAR2 (48)	操作系统程序名称
TYPE	VARCHAR2 (10)	会话类型
SQL_ADDRESS	VARCHAR2 (20)	与SQL_HASH_VALUE一起使用，以标识当前正在执行的SQL语句
SQL_HASH_VALUE	NUMERIC	与SQL_ADDRESS一起使用，标识当前正在执行的SQL语句
SQL_ID	VARCHAR2 (13)	当前正在执行的SQL语句的SQL标识符
SQL_CHILD_NUMBER	NUMERIC	当前正在执行的SQL语句的子编号
PREV_SQL_ADDR	VARCHAR2 (20)	与PREV_HASH_VALUE一起使用，以标识最后执行的SQL语句

名称	类型	描述
PREV_HASH_VALUE	NUMERIC	与SQL_HASH_VALUE一起使用，标识最后执行的SQL语句
PREV_SQL_ID	VARCHAR2 (13)	最后执行的SQL语句的SQL标识符
PREV_CHILD_NUMBER	NUMERIC	最后执行的SQL语句的子编号
MODULE	VARCHAR2 (48)	通过调用DBMS_APPLICATION_INFO设置的当前执行模块的名称。SET_MODULE过程
MODULE_HASH	NUMERIC	MODULE列的哈希值
ACTION	VARCHAR2 (32)	通过调用DBMS_APPLICATION_INFO设置的当前执行操作的名称。SET_ACTION过程
ACTION_HASH	NUMERIC	ACTION列的哈希值
CLIENT_INFO	VARCHAR2 (64)	由DBMS_APPLICATION_INFO.conf设置的信息。SET_CLIENT_INFO过程
FIXED_TABLE_SEQUENCE	NUMERIC	它包含一个数字，每次会话完成对数据库的调用，并且从动态性能表进行中间选择时，该数字都会增加。性能监视器可使用此列监视数据库中的统计信息。每次性能监控器查看数据库时，它只需要查看当前活动的会话或此列中的值高于性能监控器上次看到的最大值的会话。自性能监控器最后一次查看数据库以来，所有其他会话都处于空闲状态
ROW_WAIT_OBJ#	NUMERIC	包含row_wait_row#中指定的行的表的对象ID
ROW_WAIT_FILE#	NUMERIC	包含row_wait_row#中指定的行的数据文件的标识符。只有当会话当前正在等待另一个事务提交且row_wait_obj#的值不是-1时，此列才有效
ROW_WAIT_BLOCK#	NUMERIC	包含row_wait_row#中指定的行的块的标识符。只有当会话当前正在等待另一个事务提交且row_wait_obj#的值不是-1时，此列才有效
ROW_WAIT_ROW#	NUMERIC	当前行被锁定。只有当会话当前正在等待另一个事务提交且row_wait_obj#的值不是-1时，此列才有效
LOGON_TIME	DATE	登录时间

名称	类型	描述
LAST_CALL_ET	NUMERIC	如果会话STATUS当前为ACTIVE，则该值表示自会话变为活动状态以来经过的时间(以秒为单位)。如果会话STATUS当前为INACTIVE，则该值表示自会话变为非激活状态以来经过的时间(以秒为单位)
PDML_ENABLED	VARCHAR2(3)	此列已被PDML_STATUS列取代
FAILOVER_TYPE	VARCHAR2(13)	指示是否以及在多大程度上为会话启用透明应用程序故障转移(TAF)，取值如下所示 <ul style="list-style-type: none"> • NONE - 此会话禁用故障转移 • SESSION - 客户端可以在断开连接后对其会话进行故障转移 • SELECT - 客户端也可以对正在进行的查询进行故障转移
FAILOVER_METHOD	VARCHAR2(10)	指示会话的透明应用程序故障转移方法，取值如下所示 <ul style="list-style-type: none"> • NONE - 此会话禁用故障转移 • BASIC - 客户端本身在断开连接后重新连接 • PRECONNECT - 备份实例可以支持来自它所备份的每个实例的所有连接
FAILED_OVER	VARCHAR2(3)	指示会话是否以故障转移模式运行，是否发生了故障转移(YES) (NO)
RESOURCE_CONSUMER_GROUP	VARCHAR2(32)	会话的当前资源使用者组的名称
PDML_STATUS	VARCHAR2(8)	如果ENABLED，则会话处于PARALLEL DML启用模式。如果DISABLED，则会话不支持PARALLEL DML启用模式。如果FORCED，则会话已更改为强制PARALLEL DML
PDDL_STATUS	VARCHAR2(8)	如果ENABLED，会话处于PARALLEL DDL启用模式。如果DISABLED，则会话不支持PARALLEL DDL启用模式。如果FORCED，则会话已被更改为强制PARALLEL DDL
PQ_STATUS	VARCHAR2(8)	如果ENABLED，会话处于PARALLEL QUERY启用模式。如

名称	类型	描述
		果DISABLED，则会话不支持PARALLEL QUERY启用模式。如果FORCED，则会话已被更改为强制PARALLEL QUERY
CURRENT_QUEUE_DURATION	NUMERIC	如果排队(1)，则表示当前会话已排队的时间。如果当前没有排队，则该值为0
CLIENT_IDENTIFIER	VARCHAR2(64)	会话的客户端标识符
BLOCKING_SESSION_STATUS	VARCHAR2(11)	<p>这一列提供了关于是否存在阻塞会话的详细信息，取值如下所示</p> <ul style="list-style-type: none"> • VALID - 有一个阻塞会话，它在BLOCKING_INSTANCE和BLOCKING_SESSION列中标识 • NO HOLDER - 没有会话阻塞这个会话 • NOT IN WAIT - 此会话不在等待状态 • UNKNOWN - 阻塞会话未知
BLOCKING_INSTANCE	NUMERIC	阻塞会话的实例标识符。只有BLOCKING_SESSION_STATUS的值为valid时，此列才有效
BLOCKING_SESSION	NUMERIC	阻断会话的会话标识。只有BLOCKING_SESSION_STATUS的值为valid时，此列才有效
SEQ#	NUMERIC	唯一标识当前或最后一次等待的数字(每次等待递增)
EVENT#	NUMERIC	如果会话当前正在等待，则表示会话正在等待的资源或事件的编号。如果会话不在等待中，则为会话最近等待的资源或事件的编号
EVENT	VARCHAR2(64)	如果会话当前正在等待，则会话正在等待的资源或事件。如果会话不在等待中，则会话最近等待的资源或事件
P1TEXT	VARCHAR2(64)	第一个等待事件参数的描述
P1	NUMERIC	第一个等待事件参数(十进制)
P1RAW	VARCHAR2(20)	第一个等待事件参数(十六进制) P1RAW列显示的值与P1列相同，只是数字以十六进制形式显示
P2TEXT	VARCHAR2(64)	第二个等待事件参数的描述
P2	NUMERIC	第二个等待事件参数(十进制)

名称	类型	描述
P2RAW	VARCHAR2 (20)	第二个等待事件参数(十六进制), P2RAW列显示的值与P2列相同, 只是数字以十六进制形式显示
P3TEXT	VARCHAR2 (64)	第三个等待事件参数的描述
P3	NUMERIC	第三个等待事件参数(十进制)
P3RAW	VARCHAR2 (20)	第三个等待事件参数(十六进制) “P3RAW” 列显示的值与 “P3” 列相同, 不同之处在于数值以十六进制形式显示
WAIT_CLASS_ID	NUMERIC	等待事件类的标识符
WAIT_CLASS#	NUMERIC	等待事件的类的编号
WAIT_CLASS	VARCHAR2 (64)	等待事件的类的名称
WAIT_TIME	NUMERIC	<p>如果会话当前处于等待状态, 则该值为0。如果会话没有处于等待状态, 则取值如下所示</p> <ul style="list-style-type: none"> • > 0 - 值是最后一次等待的持续时间, 以百分之一秒为单位 • -1 - 最后一次等待的持续时间小于百分之一秒 • -2 - 参数TIMED_STATISTICS 设置为false <p>该列已被弃用, 取而代之的是WAIT_TIME_MICRO和STATE列</p>
SECONDS_IN_WAIT	NUMERIC	<p>如果会话当前正在等待, 则该值为当前等待的等待时间。如果会话没有处于等待状态, 则该值是自上次等待开始以来的时间。 该列已被弃用, 取而代之的是WAIT_TIME_MICRO和TIME_SINCE_LAST_WAIT_MICRO</p>
STATE	VARCHAR2 (19)	<p>等待状态, 取值如下所示</p> <ul style="list-style-type: none"> • WAITING - 会话当前正在等待 • waiting UNKNOWN TIME - 最后一次等待的持续时间未知;这是参数TIMED_STATISTICS设置为false时的值 • WAITED SHORT TIME - 上次等待不到百分之一秒

名称	类型	描述
		<ul style="list-style-type: none"> waiting KNOWN TIME -最后一次等待的时间在WAIT_TIME列中指定
SERVICE_NAME	NUMERIC	会话的服务名称
SQL_TRACE	VARCHAR2 (8)	指示SQL跟踪是启用(enabled)还是禁用(disabled)
SQL_TRACE_WAITS	VARCHAR2 (5)	指示是否启用等待跟踪(TRUE)或不启用(FALSE)
SQL_TRACE_BINDS	VARCHAR2 (5)	指示是否启用绑定跟踪(TRUE)或不启用(FALSE)

1. 171. V\$SYSSTAT

V\$SYSSTAT视图描述当前系统统计信息。

表 1. 172. V\$SYSSTAT的列

名称	类型	描述
STATISTIC#	NUMBER	统计数字，注意：统计数字不能保证从一个版本到另一个版本保持不变。因此，在您的应用程序中，您应该依赖统计名称而不是统计数字
NAME	VARCHAR2 (63)	统计的名字。你可以通过查询V\$STATNAME视图得到一个完整的统计名称列表
VALUE	NUMBER	统计值

第 2 章 数据目录

在做任何事情之前，必须先初始化磁盘上的数据存储区，叫作数据库集群。一个数据库集群是一系列数据库的集合，这些数据库可以通过单个数据库服务器的实例管理。在初始化后，一个数据库集群将包含一个叫uxdb的数据库，这个库是给工具、用户和第三程序使用的缺省数据库。数据库服务器本身并不要求uxdb数据库的存在，但是很多外部工具假设它存在。另外一个在每个集群初始化过程中创建的数据库叫template1。正如其名一样，这个数据库将作为随后创建的数据库的模版；在实际工作中不应该使用这个库。用文件系统的术语来说，一个数据库集群是一个目录，所有数据都将存放在这个目录中，称做数据目录或数据区。数据存放在哪里完全取决于用户的选择，没有缺省值。初始化一个数据库集群，可以使用initdb命令，这个命令与UXDB一起安装。UXDB的数据目录可以用-D 选项指定为本地目录，也可以通过-Z选项指定为分布式文件系统目录。数据库集群所需要的配置和数据文件都存储在集群的数据目录里，该数据目录通常记录在环境变量UXDATA中。

UXDATA目录包含几个子目录以及一些控制文件，如表 2.1 “UXDATA的内容”所示。除了这些必要的东西之外，集群的配置文件uxsinodb.conf、ux_hba.conf和ux_ident.conf通常都存储在UXDATA中，不过可以把它们放在其他位置。

表 2.1. UXDATA的内容

项	描述
UX_VERSION	一个包含UXDB主版本号的文件
base	包含每个数据库对应的子目录的子目录
current_logfiles	记录日志记录收集器当前写入的日志文件
global	包含集群范围的表的子目录，比如lux_database
ux_commit_ts	包含事务提交时间戳数据的子目录
ux_dynshmem	包含被动态共享内存子系统所使用的文件的子目录
ux_logical	包含用于逻辑复制的状态数据的子目录
ux_multixact	包含多事务（multi-transaction）状态数据的子目录（用于共享的行锁）
ux_notify	包含LISTEN/NOTIFY状态数据的子目录
ux_replslot	包含复制槽数据的子目录
ux_serial	包含已提交的可序列化事务信息的子目录
ux_snapshots	包含导出的快照的子目录
ux_stat	包含用于统计子系统的永久文件的子目录
ux_stat_tmp	包含用于统计信息子系统的临时文件的子目录
ux_subtrans	包含子事务状态数据的子目录
ux_tblspc	包含指向表空间的符号链接的子目录
ux_twophase	包含用于预备事务状态文件的子目录
ux_wal	包含 WAL （预写日志）文件的子目录
ux_xact	包含事务提交状态数据的子目录
uxsinodb.auto.conf	一个用于存储由ALTER SYSTEM 设置的配置参数的文件

项	描述
uxmaster.opts	一个记录服务器最后一次启动时使用的命令行参数的文件
uxmaster.pid	一个锁文件，记录着当前的 uxmaster 进程 ID (PID)、集群数据目录路径、uxmaster 启动时间戳、端口号、Unix 域套接字目录路径 (Windows 上为空)、第一个可用的 listen_address (IP 地址或者*, 或者为空表示不在 TCP 上监听) 以及共享内存段 ID (服务器关闭后该文件不存在)

第 3 章 日志文件

3.1. ux_log

ux_log是数据库活动日志。在\$UXDATA/uxsinodb.conf中可以对日志进行相应的修改。下面介绍常用的日志配置。

表 3.1. ux_log常用日志配置

参数	默认值	描述
logging_collector	on	是否将日志重定向至文件中，DB启动过程中会将日志重定向\$UXDATA/ux_log中。该配置修改后，需要重启DB服务
log_directory	UXDATA的相对路径，即\$UXDATA/ux_log	日志文件目录，可以修改为绝对路径。例如将此配置修改为/var/log/ux_log下，必须先创建此目录，并修改权限。
log_filename	UXDB-%Y-%m-%d_%H%M%S.log	日志文件命名形式，使用默认即可。
log_rotation_age	1d	单个日志文件的生存期，默认1天，在日志文件大小没有达到log_rotation_size时，一天只生成一个日志文件。
log_rotation_size	10MB	单个日志文件的大小，如果时间没有超过log_rotation_age，一个日志文件最大只能到10M，否则将新生成一个日志文件。
log_truncate_on_rotation	off	当日志文件已存在时，该配置如果为off，新生成的日志将在文件尾部追加，如果为on，则会覆盖原来的日志。
log_lock_waits	off	控制当一个会话等待时间超过deadlock_timeout而被锁时是否产生一个日志信息。在判断一个锁等待是否会影响性能时是有用的。
log_statement	none	控制记录哪些SQL语句。none：不记录；ddl：记录所有数据定义命令，比如CREATE，ALTER，和DROP 语句；mod：记录所有ddl语句，加上数据修改语句INSERT，UPDATE等；all：记录所有执行的语句，跟踪整个数据库执行的SQL语句。
log_duration	off	记录每条SQL语句执行完成消耗的时间，将此配置设置为on，

参数	默认值	描述
		用于统计哪些SQL语句耗时较长。
log_min_duration_statement	-1	-1: 不可用; 0: 将记录所有SQL语句和它们的耗时; >0: 只记录那些耗时超过(或等于)这个值(ms)的SQL语句。适用于跟踪耗时较长, 可能存在性能问题的SQL语句。使用log_statement和log_duration也能够统计SQL语句及耗时, 但是SQL语句和耗时统计结果可能相差很多行, 或在不同的文件中, 但是log_min_duration_statement会将SQL语句和耗时在同一行记录, 更方便阅读。
log_connections	off	是否记录连接日志。
log_disconnections	off	是否记录连接断开日志。
log_line_prefix	%m %p %u %d %r	日志输出格式。配置文件解释%m, %p的实际意义。能够记录时间, 用户名称, 数据库名称, 客户端IP和端口, 方便定位问题。
log_timezone	Asia/Shanghai	日志时区, 最好和服务器设置同一个时区, 方便问题定位。

3.2. ux_xlog

ux_xlog记录UXDB的WAL信息, 也就是事务日志信息, 默认单个大小是16M。这些信息通常记录在名称类似'0000000100000000000000013'这样的文件中, 这些日志会在定时回滚恢复(PITR), 流复制(Replication Stream)以及归档时被用到。这些日志是非常重要的, 记录着数据库发生的各种事务信息, 不得随意删除或者移动这类日志文件, 否则数据库会有无法恢复的风险。当归档或者流复制发生异常时, 事务日志会不断地生成, 有可能会造成磁盘空间被塞满, 最终导致UXDB挂掉或者无法正常启动。遇到这种情况, 可以先关闭归档或者流复制功能; 备份ux_xlog日志到其他路径, 但不要删除; 删除较早时间的ux_xlog, 有一定空间后再试着启动UXDB。

3.2.1. 设置

wal_level (enum)

wal_level决定多少信息写入到 WAL 中。默认值是**replica**, 它写入足够的数据以支持WAL归档和复制, 包括在备用服务器上运行只读查询。 **minimal**删除除了从崩溃或立即关闭中恢复所需的信息之外的所有日志记录。最后, **logical**会增加支持逻辑解码所需的信息。每个层次包括所有更低层次 记录的信息。这个参数只能在服务器启动时设置。

在**minimal**级别中, 某些批量操作的 WAL 日志可以被安全地跳过, 这可以使那些操作更快。这种优化可以应用的操作包括:

CREATE TABLE AS

CREATE INDEX CLUSTER

COPY 在同一个事务中被创建或清空的表中

但最少的 WAL 不会包括足够的信息来从基础备份和 WAL 日志中重建数据，因此，要启用 WAL 归档 ([archive_mode](#)) 和流复制，必须使用 **replica** 或更高级别。

在 **logical** 层，与 **replica** 相同的信息会被记录，外加上 允许从 WAL 抽取逻辑修改集所需的信息。使用级别 **logical** 将增加 WAL 容量，特别是如果为了 **REPLICA IDENTITY FULL** 配置了很多表并且执行了很多 **UPDATE** 和 **DELETE** 语句时。

此参数还允许值 **archive** 和 **hot_standby**。这些仍然被接受， 但映射到 **replica**。

fsync (boolean)

如果打开这个参数，UXDB服务器将尝试确保更新被物理地写入到磁盘，做法是发出 **fsync()** 系统调用或者使用多种等价的方法（见 [wal_sync_method](#)）。这保证了数据库集群在一次操作系统或者硬件崩溃后能恢复到一个一致的状态。

虽然关闭 **fsync** 常常可以得到性能上的收益，但当发生断电或系统崩溃时可能造成不可恢复的数据损坏。因此，只有在能很容易地从外部数据中重建整个数据库时才建议关闭 **fsync**。

能安全关闭 **fsync** 的环境的例子包括从一个备份文件中初始加载一个新数据库集群、使用一个数据库集群来在数据库被删掉并重建之后处理一批数据，或者一个被经常重建但却不用于失效备援的只读数据库克隆。单独的高质量硬件不足以成为关闭 **fsync** 的理由。

当把 **fsync** 从关闭改成打开时，为了可靠的恢复，需要强制在内核中的所有被修改的缓冲区进入持久化存储。这可以在多个时机来完成：在集群被关闭时或在 **fsync** 因为运行 **initdb --sync-only** 而打开时、运行 **sync** 时、卸载文件系统时或者重启服务器时。

在很多情况下，为不重要的事务关闭 [synchronous_commit](#) 可以提供很多关闭 **fsync** 的潜在性能收益，并不会有的同时， 关闭 **fsync** 可以提供很多潜在的性能优势，而不会有伴随着的数据损坏风险。

fsync 只能在 **uxsinodb.conf** 文件中或在服务器命令行上设置。如果你关闭这个参数，请也考虑关闭 [full_page_writes](#)。

synchronous_commit (enum)

指定在命令返回 “success” 指示给客户端之前，一个事务是否需要等待 WAL 记录被写入磁盘。合法的值是 **on**、**remote_apply**、**remote_write**、**local** 和 **off**。默认的并且安全的设置是 **on**。当设置为 **off** 时，在向客户端报告成功和真正保证事务不会被服务器崩溃威胁之间会有延迟（最大的延迟是 [wal_writer_delay](#) 的三倍）。不同于 **fsync**，将这个参数设置为 **off** 不会产生数据库不一致性的风险：一个操作系统或数据库崩溃可能会造成一些最近据说已提交的事务丢失，但数据库状态是一致的，就像这些事务已经被中止。因此，当性能比完全确保事务的持久性更重要时，关闭 **synchronous_commit** 可以作为一个有效的代替手段。

如果 **synchronous_standby_names** 为非空，这个参数也控制事务提交是否将等待事务的 WAL 记录被复制到后备服务器上。当这个参数被设置为 **on** 时，直到来自于当前同步的后备服务器的回复指示该后备服务器已经收到了事务的提交记录并将其刷入了磁盘，主服务器上的事务才会提交。这保证事务将不会被丢失，除非主服务器和后备服务器都遭受到了数据库存储损坏的问题。 当设置为 **remote_apply** 时，提交将等待， 直到来自当前同步备用数据库的回复表明它们已收到事务的提交记录并应用该事务， 以便它对备用数据库上的查询可见。 当这个参数被设置为 **remote_write** 时，提交将等待，直到来自当前同步的后备服务器的回复指示该服务器已经收到了该事务的提交记录并且已经把该记录写出到后备服务器的操作

系统，这种设置足以保证数据在后备服务器的UXDB实例崩溃时得以保存，但是不能保证后备服务器遭受操作系统级别崩溃时数据能被保存，因为数据不一定必须要在后备机上达到稳定存储。最后，设置`local`会导致提交等待本地刷写到磁盘而不是复制完成。使用同步复制时通常不能达到想要的效果，但是为了完整性，仍然提供了该选项。

如果`synchronous_standby_names`为空，设置`on`、`remote_apply`、`remote_write`和`local`都提供了同样的同步级别：事务提交只等待本地刷写磁盘。

这个参数可以随时被修改；任何一个事务的行为由其提交时生效的设置决定。因此，可以同步提交一些事务，同时异步提交其他事务。例如，当默认是相反时，实现一个单一多语句事务的异步提交，在事务中发出`SET LOCAL synchronous_commit TO OFF`。

`wal_sync_method` (enum)

用来向强制 WAL 更新到磁盘的方法。如果`fsync`是关闭的，那么这个设置就不相关，因为 WAL 文件更新将根本不会被强制。可能的值是：

- `open_datasync`（用`open()`选项`O_DSYNC`写 WAL 文件）
- `fdatsync`（在每次提交时调用`fdatsync()`）
- `fsync`（在每次提交时调用`fsync()`）
- `fsync_writethrough`（在每次提交时调用`fsync()`，强制任何磁盘写高速缓存的直通写）
- `open_sync`（用`open()`选项`O_SYNC`写 WAL 文件）

`open_*` 选项也可以使用`O_DIRECT`（如果可用）。不是在所有平台上都能使用所有这些选择。默认值是列表中第一个被平台支持的那个，不过`fdatsync`是 Linux 中的默认值。默认值不一定是理想的；有可能需要修改这个设置或系统配置的其他方面来创建一个崩溃-安全的配置，或达到最佳性能。这个参数只能在`uxsinodb.conf`文件中或在服务器命令行上设置。

`full_page_writes` (boolean)

当这个参数为打开时，UXDB服务器在一个检查点之后的页面的第一次修改期间将每个页面的全部内容写到 WAL 中。这么做是因为在操作系统崩溃期间正在处理的一次页写入可能只有部分完成，从而导致在一个磁盘页面中混合有新旧数据。在崩溃后的恢复期间，通常存储在 WAL 中的行级改变数据不足以完全恢复这样一个页面。存储完整的页面映像可以保证页面被正确存储，但代价是增加了必须被写入 WAL 的数据量（因为 WAL 重放总是从一个检查点开始，所以在检查点后每个页面的第一次改变时这样做就够了。因此，一种减小全页面写开销的方法是增加检查点间隔参数值）。

把这个参数关闭会加快正常操作，但是在系统失败后可能导致不可恢复的数据损坏，或者静默的数据损坏。其风险类似于关闭`fsync`，但是风险较小。并且只有在可关闭`fsync`的情况下才应该关闭它。

关闭这个选项并不影响用于时间点恢复（PITR）的 WAL 归档使用。

这个参数只能在`uxsinodb.conf`文件中或在服务器命令行上设置。默认值是`on`。

`wal_log_hints` (boolean)

当这个参数为`on`时，UXDB服务器一个检查点之后页面被第一次修改期间把该磁盘页面的整个内容都写入 WAL，即使对所谓的提示位做非关键修改也会这样做。

如果启用了数据校验和，提示位更新总是会被 WAL 记录并且这个设置会被忽略。你可以使用这个 设置测试如果你的数据库启用了数据校验和，会有多少额外的 WAL 记录发生。

这个参数只能在服务器启动时设置。默认值是off。

wal_compression (boolean)

当这个参数为on， 如果full_page_writes为打开或者处于基础备份期间， UXDB服务器 会压缩写入到 WAL 中的完整页面镜像。压缩页面镜像将在 WAL 重放时 被解压。默认值为off。只有超级用户可以更改这个设置。

打开这个参数可以减小 WAL 所占的空间且无需承受不可恢复的数据损坏风险， 但是代价是需要额外的 CPU 开销以便在 WAL 记录期间进行压缩以及在 WAL 重放时解压。

wal_buffers (integer)

用于还未写入磁盘的 WAL 数据的共享内存量。默认值 -1 选择等于shared_buffers的 1/32 的尺寸（大约3%），但是不小于64kB也不大于 WAL 段的尺寸（通常为16MB）。如果自动的选择太大或太小可以手工设置该值，但是任何小于32kB的正值都将被当作32kB。 如果指定值时没有单位，则以 WAL 块作为单位，即为 XLOG_BLCKSZ 字节，通常为8kB。这个参数只能在服务器启动时设置。

在每次事务提交时，WAL 缓冲区的内容被写出到磁盘，因此极大的值不可能提供显著的收益。不过，把这个值设置为几个兆字节可以在一个繁忙的服务器（其中很多客户端会在同一时间提交）上提高写性能。由默认设置 -1 选择的自动调节将在大部分情况下得到合理的结果。

wal_writer_delay (integer)

指定WAL写入器刷写WAL的频繁程度，以时间为单位。 在刷写WAL之后，写入器将根据wal_writer_delay所给出的时间长度进行睡眠，除非被一个异步提交的事务提前唤醒。 如果最近的刷写发生在wal_writer_delay之前，并且小于 wal_writer_flush_after WAL的值产生之后，那么WAL只会被写入操作系统，而不会被刷写到磁盘。 如果指定值时没有单位，则以毫秒作为单位。 默认值是 200 毫秒（200ms）。注意在很多系统上，有效的睡眠延迟粒度是 10 毫秒，把wal_writer_delay设置为一个不是 10 的倍数的值，其效果和把它设置为大于该值的下一个 10 的倍数产生的效果相同。这个参数只能在uxsinodb.conf文件中或在服务器命令行上设置。

wal_writer_flush_after (integer)

指定 WAL 写入器刷写 WAL 的频繁程度，以卷为单位。 如果最近的刷写发生在wal_writer_delay之前，并且小于wal_writer_flush_afterWAL的值产生之后，那么WAL只会被写入操作系统，而不会被刷写到磁盘。 如果wal_writer_flush_after被设置为0，则WAL数据总是会被立即刷写。 如果指定值时没有单位，则以WAL块作为单位，即为XLOG_BLCKSZ字节，通常为8kB。默认是1MB。这个参数只能在uxsinodb.conf文件中或者服务器命令行上设置。

commit_delay (integer)

在一次 WAL 刷写被发起之前，commit_delay增加一个时间延迟。如果系统负载足够高，使得在一个给定间隔内有额外的事务准备好提交，那么通过允许更多事务通过一个单次 WAL 刷写来提交能够提高组提交的吞吐量。但是，它也把每次 WAL 刷写的潜伏期增加到了最多commit_delay。因为如果没有其他事务准备好提交，就会浪费一次延迟，只有在当一次刷写将要被发起时有至少commit_siblings个其他活动事务时，才会执行一次延迟。另外，如

果fsync被禁用，则将不会执行任何延迟。如果指定值时没有单位，则以微秒作为单位。默认的commit_delay是零（无延迟）。只有超级用户才能修改这个设置。

第一个准备好刷写的进程会等待配置的间隔，而后续的进程只等到领先者完成刷写操作。

commit_siblings (integer)

在执行commit_delay延迟时，要求的并发活动事务的最小数目。大一些的值会导致在延迟间隔期间更可能有至少另外一个事务准备好提交。默认值是五个事务。

3.2.2. 检查点

checkpoint_timeout (integer)

自动 WAL 检查点之间的最长时间。如果指定值时没有单位，则以秒为单位。合理的范围在30秒和1天之间。默认是 5 分钟（5min）。增加这个参数的值会增加崩溃恢复所需的时间。这个参数只能在uxsinodb.conf文件中或在服务器命令行上设置。

checkpoint_completion_target (floating point)

指定检查点完成的目标，作为检查点之间总时间的一部分。默认是0.5。这个参数只能在uxsinodb.conf文件中或在服务器命令行上设置。

checkpoint_flush_after (integer)

当执行检查点时写入的数据量超过此数量时，就尝试强制 OS 把这些写发送到底层存储。这样做将会限制内核页面高速缓存中的脏数据数量，降低在检查点末尾发出fsync或者 OS 在后台大批量写回数据时被卡住的可能性。那常常会导致大幅度压缩的事务延迟，但是也有一些情况（特别是负载超过shared_buffers但小于 OS 页面高速缓存）的性能会降低。这种设置可能会在某些平台上没有效果。如果指定值时没有单位，则以块为单位，即为BLCKSZ 字节，通常为8kB。合法的范围在0（禁用强制写回）和2MB之间。Linux 上的默认值是256kB，其他平台上是0（如果BLCKSZ不是8kB，则默认值和最大值会按比例缩放放到它）。这个参数只能在uxsinodb.conf文件中或者服务器命令行上设置。

checkpoint_warning (integer)

如果由于填充WAL段文件导致的检查点之间的间隔低于这个参数表示的时间量，那么就向服务器日志写一个消息（它建议增加max_wal_size的值）。如果指定值时没有单位，则以秒为单位。默认值是 30 秒（30s）。零则关闭警告。如果checkpoint_timeout低于checkpoint_warning，则不会有警告产生。这个参数只能在uxsinodb.conf文件中或在服务器命令行上设置。

max_wal_size (integer)

在自动 WAL 检查点之间允许 WAL 增长到的最大尺寸。这是一个软限制，在特殊的情况下 WAL 尺寸可能会超过 max_wal_size，例如在重度负荷下、archive_command失败或者高的wal_keep_segments的设置。如果指定值时没有单位，则以兆字节为单位。默认为1GB。增加这个参数 可能导致崩溃恢复所需的时间。这个参数只能在uxsinodb.conf文件或者服务器命令行上设置。

min_wal_size (integer)

只要 WAL 磁盘用量保持在这个设置之下，在检查点时旧的 WAL 文件总是 被回收以便未来使用，而不是直接被删除。这可以被用来确保有足够的 WAL 空间被保留来应付 WAL 使用的高

峰，例如运行大型的批处理任务。如果指定值时没有单位，则以兆字节为单位。默认是 80 MB。这个参数只能在`uxsinodb.conf`文件或者 服务器命令行上设置。

3.2.3. 归档

`archive_mode` (enum)

当启用`archive_mode`时，可以通过设置`archive_command`命令将完成的 WAL段发送到归档存储。除用于禁用的`off`之外，还有两种模式`on`和 `always`。在普通操作期间，这两种模式之间没有区别，但是当设置为`always`时，WAL 归档器在归档恢复 或者后备模式下也会被启用。在`always`模式下，所有从归档恢复 的或者用流复制传来的文件将被（再次）归档。

`archive_mode`和`archive_command` 是独立的变量，这样可以在不影响归档模式的前提下修改`archive_command`。这个参数只能在服务器启动时设置。当`wal_level`被设置为`minimal`时，`archive_mode`不能被启用。

`archive_command` (string)

本地 shell 命令被执行来归档一个完成的 WAL 文件段。字符串中的任何`%p`被替换成要被归档的文件的路径名，而`%f`只被文件名替换（路径名是相对于服务器的工作目录，即集群的数据目录）。如果要在命令里嵌入一个真正的`%`字符，可以使用`%%`。该命令只在成功时返回一个零作为退出状态。

这个参数只能在`uxsinodb.conf`文件中或在服务器命令行上设置。除非服务器启动时启用了`archive_mode`，否则它会被忽略。如果`archive_mode`被启用时，`archive_command`是一个空字符串（默认），WAL 归档会被临时禁用，但服务器仍会继续累计 WAL 段文件，期待着一个命令被提供。将`archive_command`设置为一个只返回真但不做任何事的命令（例如`/bin/true`或 Windows 上的`REM`）实际上会禁用归档，也会打破归档恢复所需的 WAL 文件链，因此只有在极少数情况下才能用。

`archive_timeout` (integer)

`archive_command`仅在已完成的 WAL 段上调用。因此，如果服务器只产生很少的 WAL 流量（或产生流量的周期很长），那么在事务完成和它被安全地记录到归档存储之间将有一个很长的延迟。为了限制未归档数据存在的时间，可以设置`archive_timeout`来强制服务器来周期性地切换到一个新的 WAL 段文件。当这个参数被设置为大于零时，只要从上次段文件切换后过了参数所设置的时间量，并且已经有过任何数据库活动（包括一个单一检查点），服务器将切换到一个新的段文件（如果没有数据库活动则会跳过检查点）。注意，由于强制切换而提早关闭的被归档文件仍然与完整的归档文件长度相同。因此，使用非常短的`archive_timeout`是不明智的——它将占用巨大的归档存储。一分钟左右的`archive_timeout`设置通常比较合理。如果希望数据能被更快地从主服务器上复制下来，应该考虑使用流复制而不是归档。如果指定值时没有单位，则以秒为单位。这个参数只能在`uxsinodb.conf`文件中或在服务器命令行上设置。

3.3. ux_clog

`ux_clog`文件也是事务日志文件，但与`ux_xlog`不同的是它记录的是事务的元数据(metadata)，记录哪些事务已经完成，哪些事务没有完成。这个日志文件一般非常小，但重要性相当高，不得随意删除或者对其更改信息。