ICS 35.020
L 70

团　　体　　标　　准

T/GDCSA XXX-XXXX

关系型数据库国产化替换迁移技术标准

Technical standard for domestication replacement and migration of relational databases

（征求意见稿）

2025-XX-XX 发布　　2025-XX-XX 实施

XXXXXXXXX　　发 布

目　　次

前　言

本文件按照GB/T 1.1—2020《标准化工作导则 第1部分：标准化文件的结构和起草规则》的规定起草。

本文件由中国第一汽车集团有限公司和中电科金仓（北京）科技股份有限公司联合提出。

本文件由北京网络空间安全协会和广东省网络空间安全协会归口。

本文件起草单位：中国第一汽车集团有限公司、中电科金仓（北京）科技股份有限公司、广东新兴国家网络安全与信息化发展研究院、网安联认证中心有限公司、广州华南检验检测中心有限公司和国源天顺科技产业集团有限公司。

本文件主要起草人：门欣、陈韵、冷健全、肖飞、王建华、何嗣华、张俊余、刘延明、董铭民。

关系型数据库国产化替换迁移技术标准

1 范围

本文件规定了国产关系型数据库的迁移流程、调研评估、迁移准备、迁移实施、测试验证和服务保障等要求。

本文件适用于关系型数据库国产化替换迁移的研发、测试、评估、应用、运维和管理。

2 规范性引用文件

本文件没有规范性引用文件。

3 术语和定义

下列术语和定义适用于本文件。

3.1

关系型数据库 relational database

主流关系型数据库包括 MySQL、Oracle、SQL Server、PostgreSQL 等流行的国外关系型数据库，默认关系型数据库需要进行国产替换的目标。

3.2

国产关系型数据库 domestic relational database

国产关系型数据库产品包括达梦数据库管理系统、金仓数据库管理系统等国产数据库产品。

3.3

归档日志 archive logs

当条件满足时，关系型数据库在线重做日志以文件形式保存到硬盘（持久化）。关系型数据库归档日志的存在极大地方便了关系型数据库备份、恢复、管理使用。

3.4

全量迁移 full migration

当前需要迁移的数据库系统的全量数据迁移、同步工作。

3.5

增量数据 incrementing data

在数据库系统迁移过程中，新产生的数据即为增量数据，这些数据直接保存到数据库系统。

4 迁移流程

国产关系型数据库迁移流程见图1，包括调研评估、迁移准备、迁移实施、测试验证和服务保障5个阶段。

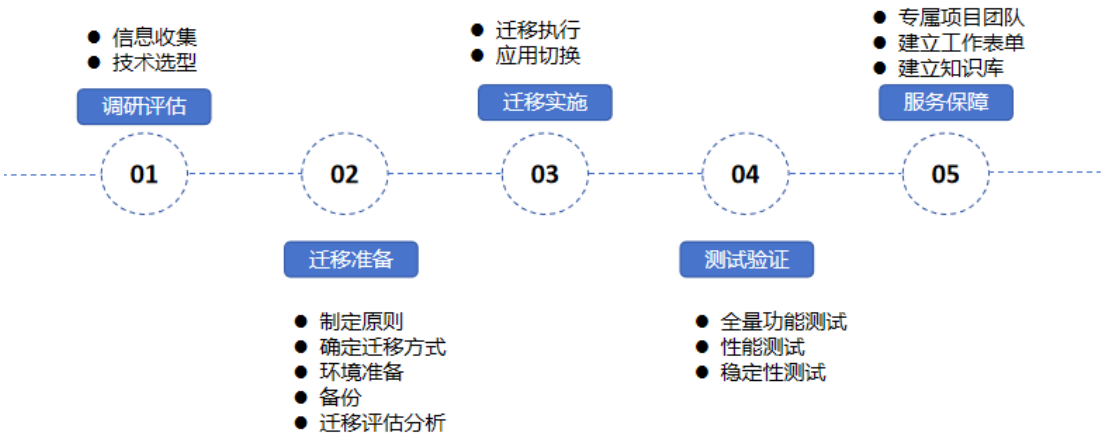


图 1 迁移工作流程

国产关系型数据库系统迁移参与方应包括：

a) 用户方：全面组织和协调国产关系型数据库迁移工作，负责组织开展调研评估，明确迁移工作所涉及的业务软件、操作系统环境、网络权限、硬件环境、人员等资源，协助用户进行调研评估，负责对迁移准备、实施，并对迁移工作效果进行检查；

b) 关系型数据库厂商：负责对关系数据库国产化迁移过程、版本更新等工作进行服务保障；

c) 应用服务厂商：协助完成迁移工作中的调研评估、业务系统运行和服务保障等工作；

d) 测试机构：国产关系型数据库迁移实施后，负责国产关系型数据库能力和业务系统可行性测试验证。

5 调研评估

5.1 业务环境调研

全面梳理企业内信息化业务系统，最终形成迁移工作台账，应包括但不限于以下盘点信息：

a) 基础设施：服务器品牌型号、硬件配置以及操作系统、数据库、中间件、云计算等基础软件名称和版本号；

b) 开发/运行环境：业务系统的部署架构、关键组件包及依赖关系、源仓库配置；

c) 业务系统属性：业务系统名称、业务系统的自主程度、重要等级、开发年限、业务使用的数据及存放位置等。

业务环境调研表参见附录A。

5.2 关系型数据库技术路线选型评估

国产关系型数据库技术路线选型应满足以下要求：

a) 数据库兼容性：对于不同版本关系型数据库可提供国产化迁移能力；

b) 环境兼容性：对企业各种存量信息化资源进行兼容，至少包括中间件、操作系统、硬件、第三方主流开发框架、不同类型的关系型数据库生态工具兼容能力；

c） 定制能力：对于无法迁移的版本，需提供国产关系型数据库定制版本；
d） 合规性：通过安全可靠测评要求；
e） 技术支持：关系型数据库产品厂商能够提供新增功能定制开发服务。

6 迁移准备

6.1 迁移原则

迁移工作应遵循以下原则：
a） 优先迁移已停止技术维护服务的开源版、商用版过老的关系型数据库；
b） 严格管控增量，默认使用国产关系型数据库，特殊情况使用定制版兼容方案；
c） 遵循由易到难、适度超前，制定整体迁移计划，并分阶段对待迁系统进行排期。

6.2 迁移方式

根据业务系统在数据库架构的不同，应考虑以下两种迁移场景：
a） 原关系型数据库架构为单机：没有归档日志，因此不能选择增量数据迁移，只能选择全量迁移方式；
b） 原关系型数据库架构主备或主从架构：这种情况优先选择全量数据迁移+增迁移的方式，这种方式对业务系统的停机时间要求是最短的。

6.3 环境准备

环境准备工作包括：
a） 硬件服务器等资源准备，开通相关网络权限，包括但不限于数据迁移工具联接，国产关系型数据库、需替换的开源版、商用版源数据库端口访问权限，业务应用到国产关系型数据库的端口权限；
b） 按照源关系型数据库部署架构进行国产关系型数据库的部署工作，保证架构对等，注意部署时数据库端口的选择要与源数据库保持一致，减少应用改动工作量；
c） 部署迁移工具或者迁移服务。

6.4 备份

为了保证业务系统出现问题时可以回退恢复，在应用切换的时间点，应进行原数据库的数据备份：
a） 通过逻辑备份或者物理备份方式对源关系型数据库进行数据全量备份；
b） 所有备份恢复应在实验室环境中先行进行验证。

6.5 迁移评估分析

进行关系型数据库国产化迁移前，首先应进行当前关系型数据库和国产数据库相关信息进行对比，并对业务状况进行评估，包括但不限于：
a） 业务使用到的数据库语法，关系型数据库常用语法兼容性对比示例参见附录B.1；
b） 数据库架构类型，单机版归档日志条件是否满足，选择数据同步模式；
c） 开发框架和版本类型；
d） 国产化替换数据库的驱动变更，驱动兼容性对比示例参考附录B.1；
e） 业务的停机时间、停机窗口（建议选择在业务低峰期）、数据量大小、归档大小。

7 迁移实施

7.1 迁移方式

关系型数据库需支持以下两种迁移方式，2种迁移方式的情况分析可参见7.2章节：

a) 全量数据迁移：在数据迁移之前，业务系统需停止服务，停机时间包括数据迁移时间和应用切换时间，适用于停机时间充足的业务系统；

b) 全量数据+增量数据迁移：数据迁移阶段不用停机，在增量数据同步完成后，业务系统切换服务，停机时间仅包括应用切换时间，适用于停机时间较短的业务系统；

注：在迁移期间，业务系统不可做DDL操作，此类操作会让同步状态异常。

7.2 迁移执行

a) 迁移工具进行配置，运行迁移工具执行迁移操作。若迁移成功则进行应用验证；若迁移失败，用户和国产关系型数据库厂商应排查问题后重新执行迁移。如迁移出现的问题无法解决，则应通过迁移工具或者备份进行回退，记录问题并交付后场研究解决方案，待方案确定后再次实施迁移。

b) 数据迁移完毕后，需要将原关系型数据库的账号权限复现到新关系型数据库中，并进行验证。

7.3 应用切换

数据同步完后，应用需及时修改应用配置，将数据库连接地址指向国产关系型数据库。如果驱动不满足要求的，同时也要修改驱动包。

8 测试验证

国产关系型数据库迁移成功后，判断业务系统是否可以正式上线，应进行以下测试：

a) 全量功能测试：通过原有的业务系统测试用例对业务进行系统测试；

b) 性能测试：通过性能测试程序测试业务系统的性能是否出现明显下降的情况，宜采用并发调用接口测试工具，编制典型业务应用的测试脚本，持续运行一定时间后，观察测试得到的事务吞吐量、用户并发数、事务响应时间、错误率等数据是否正常；

c) 稳定性测试：以最大压力值测试服务器，持续一定时间，观察服务器状态和业务程序返回状态是否正常；

d) 迁移工具功能支持：如迁移工具支持数据比对功能，提供数据迁移数据比对报告。

9 服务保障

针对已经完成迁移的国产关系型数据库，在服务阶段应满足以下要求：

a) 成立专属项目服务团队，明确组织架构和问题响应机制；

b) 建立迁移工作点检表，跟踪、回顾迁移排期执行情况，统计完成迁移的国产关系型数据库的数量、版本、数据库补丁和更新时间；

c) 建立国产关系型数据库的知识库。

附 录 A
（资料性）
业务环境调研表

表A.1规定了业务环境调研表。

表A.1 业务环境调研表

调研对象	对象分类	指标名称	调研结果
基础设施	软件信息	软件类型	
		软件名称	
		软件版本	
	硬件信息	服务器类型	
		CPU 架构	
		内存容量	
		磁盘容量	
	软件信息	操作系统名称	
		操作系统版本	
		数据库名称	
		数据库版本	
		数据库部署方式	
		数据库默认参数配置	
		中间件名称	
		中间件版本	
业务系统	基本信息	业务应用系统名称	
		业务应用系统版本	
		业务应用系统级别	
		业务部门	
		后期有无明确开发需求	
		业务部门评估迁移替代优先级	
		业务使用的数据（数据量、存放位置等）	
		其他信息	
	开发/运行环境	业务应用开发语言	
		业务应用系统架构	
		业务应用系统架构（单机、B/S、云等）	
		业务应用是否为容器部署	
		业务应用关键组件包及依赖关系	
		源仓库配置	
		其他信息	

附 录 B
（资料性）
适配评估示例

附录B展示了2个关系型数据库（当前关系数据库MySQL和目标国产关系型数据库KingbaseES）在驱动兼容性方面、命令语法上的差别，基础语法基本上兼容，在开发接口、开发框架、数据库对象个别SQL迁移时需要进行改造，具体以实际测试情况为准。

B.1 数据库服务器配置推荐

	硬件配置	用户数	平均并发用户数	金仓部署
测试	2C 8G	200 用户以内	50 并发以内	单机
	4C 16G	300 用户以内	100 并发以内	单机
生产	2C 8G	200	50	一主一备
	4C 16G	300	100	一主一备
	8C 32G	500	200	一主一备
	16C 64G	1000	300	一主一备
	32C 128G	2000	500	一主一备
备注	实际硬件配比情况，需要根据具体典型的业务场景压测、人海测试或实际运行情况动态调整。			
高可用	根据实际业务的可用性要求进行，选择各节点配置要求统一，并预留一定余度，防止节点故障集群发生切换。			
读写分离	从节点只能分摊只读请求，根据读写比例换算出各个节点的实际并发数，参考上述列表进行选配。各节点配置要求统一，并预留一定余度，防止节点故障集群发生切换。			

B.2 开发接口

（一）JDBC

数据库	KingbaseES	Mysql
连接方式	加载驱动 Class.forName(“com.kingbase8.Driver”); 建立连接 Connection conn = DriverManager.getConnection("jdbc:kingbase8://localhost:54321/test", "username", “password”); 其中，localhost:54321 是 KingbaseES 服务器的地址和端口号，test 是要连接的数据库名称，username 是 KingbaseES 用户名，password 是 KingbaseES 密码。	加载驱动 Class.forName(“com.mysql.jdbc.Driver”); 建立连接 Connection conn = DriverManager.getConnection("jdbc:mysql://localhost:3306/test", "root", “password"); 其中，localhost:3306 是 MySQL 服务器的地址和端口号，test 是要连接的数据库名称，root 是 MySQL 用户名，password 是 MySQL 密码。

（二）ODBC

数据库	KingbaseES	Mysql
连接方式-windows	在 Windows 操作系统中，可以通过控制面板中的“管理工具”打开 ODBC 数据源管理器。 添加 ODBC 数据源 在 ODBC 数据源管理器中，选择“系统 DSN”选项卡，点击“添加”按钮， 选择 KingbaseES ODBC 驱动程序： KingbaseES 8.6 ODBC Driver ANSI ，点击“完成”按钮。 配置 ODBC 数据源	在 Windows 操作系统中，可以通过控制面板中的“管理工具”打开 ODBC 数据源管理器。 添加 ODBC 数据源 在 ODBC 数据源管理器中，选择“系统 DSN”选项卡，点击“添加”按钮， 选择 MySQL ODBC 驱动程序： MySQL ODBC 8.0 ANSI Driver 和 MySQL ODBC 8.0 Unicode Driver，点击“完成”按钮。 配置 ODBC 数据源
连接方式-Linux	odbcinst.ini 的配置 [KingbaseES 8.6 ODBC Driver] Description = KingbaseES 8.6 ODBC Driver for Linux Driver= /opt/Kingbase/ES/V8R6/Odbc/kdbodbcw.so Debug = 0 CommLog = 1 odbc.ini 的配置 [kingbase] Description = KingbaseES Driver = KingbaseES 8.6 ODBC Driver Trace = No TraceFile = Database = TEST Servername = localhost Username = SYSTEM Password = 123 Port = 54321 ReadOnly = No RowVersioning = No ShowSystemTables = No ShowOidColumn = No FakeOidIndex = No ConnSettings =	odbc.ini 的配置 [ODBC Data Sources] myodbc8w = MyODBC 8.0 UNICODE Driver DSN myodbc8a = MyODBC 8.0 ANSI Driver DSN [myodbc8w] Driver = /usr/local/lib/libmyodbc8w.so Description = Connector/ODBC 8.0 UNICODE Driver DSN SERVER = localhost PORT = USER = root Password = Database = test OPTION = 3 SOCKET = [myodbc8a] Driver = /usr/local/lib/libmyodbc8a.so Description = Connector/ODBC 8.0 ANSI Driver DSN SERVER = localhost PORT = USER = root Password = Database = test OPTION = 3 SOCKET =
连接方式	ConnectionString =	ConnectionString = "DRIVER={MySQL ODBC 8.0

	"DRIVER={KingbaseES 8 ODBC Driver}; DATABASE=SAMPLES; SERVER=127.0.0.1; UID=SYSTEM; PWD=MANAGER;"	Driver}; SERVER=localhost; DATABASE=test; USER=venu; PASSWORD=venu; OPTION=3;"

（三）Python

数据库	KingbaseES	Mysql
连接方式	将对应 Python 版本的 ksycopg2 驱动解压后，把 ksycopg2 文件夹放在 Python 的模块路径中，如“D:\Python27\Lib\site-packages ” 或“/usr/lib/python35/dist-packages conn= ksycopg2.connect("dbname=TEST user=SYSTEM password=123456 host=127.0.0.1 port=54321")	import mysql.connector cnx = mysql.connector.connect(user='scott', password='password', host='127.0.0.1', database='employees') cnx.close()

（四）.NET

数据库	KingbaseES	Mysql

连接方式	using System;
	using System.Data;
	using Kdbndp;
	public class TestConnection
	{
	Public static
	void Main(string[] args)
	{
	string connString = "Server=127.0.0.1;User Id=SYSTEM;
	Password=MANAGER;Database=TEST;Port=54321";
	KdbndpConnection conn = new KdbndpConnection(connString);
	conn.Open();
	}
	}
		MySql.Data.MySqlClient.MySqlConnection
		conn;
		string myConnectionString;
		myConnectionString=
		"server=127.0.0.1;uid=root;"
		+ "pwd=12345;database=test";
		try {
		conn =
		new
		MySql.Data.MySqlClient.MySqlConnection();
		conn.ConnectionString = myConnectionString;
		conn.Open();
		}

（五）PHP

数据库	KingbaseES	Mysql
连接方式-linux	1. 打开 PHP 目录下的 php.ini ，找到 extension_dir=”./” ， 修改其为 pdo_kdb.so 文件所在目录的路径。 2. 添加扩展库，在 php.ini 中找到	1. 打开 PHP 目录下的 php.ini ，找到 extension_dir=”./”，修改其为 php_mysql.so 文件所在目录的路径。 2. 添加扩展库，在 php.ini 中找到

	“;extension=*.so”（前面的“;”表示注释）。 3. 添加 extension=pdo_kdb.so ，表示将 KingbaseES 库作为扩展库供 PHP 加载。	“;extension=*.so”（前面的“;”表示注释）。 3. 添加 extension=php_mysql.so ，表示将 Mysql 库作为扩展库供 PHP 加载。
连接方式	$dsn = 'kdb:host=localhost;dbname=TEST;port=54321';	

（六）Perl

数据库	KingbaseES	Mysql

连接方式	use DBI;
	use DBD::KB;
	printf("before connect\n");
	$dbh = DBI->connect("DBI:KB:dbname=TEST;host=192.168.4.8", 'SYSTEM', 'MANAGER');
	printf("after connect\n");
	$sth = $dbh->prepare("select 123 as a, 456 as b");
	$sth->execute();
	while(@row = $sth->fetchrow_array())
	{
	print "row = @row\n";
	}
		use DBI;
		use DBD::mysql;
		my $dbh
		=
		DBI->connect(“DBI:mysql:database=myDataBase;host=localhost”,“admin”, “password”);
		#Perl statement Execute
		my $sth = $dbh->prepare(“SELECT id, name FROM myTable WHERE name = ?”);
		$sth->execute(‘John’);
		#Fetching data
		#You can perform operations like -
		while (my @data = $sth->fetchrow_array()) {
		print “@data\n”;
		}
		#Disconnect form Database
		$dbh->disconnect();

B.3 开发框架

（一）Mybatis

数据库	KingbaseES	Mysql
连接方式	在 property 中增加如下声明： jdbc.driverClassName=com.kingbase8.Driver jdbc.url=jdbc:kingbase8://localhost:54321/TEST jdbc.username= 登录名 jdbc.password= 登录密码 在 config.xml 中增加如下声明： <environments default="development">	在 property 中增加如下声明： jdbc.driverClassName=com.mysql.jdbc.Driver jdbc.url=jdbc:mysql://localhost:3306/mydatabase jdbc.username= 登录名 jdbc.password= 登录密码 在 config.xml 中增加如下声明： <environments default="development"> <environment id="development"> <transactionManager type="JDBC"/>

	<environment id="development"> <transactionManager type="JDBC"/> <dataSource type="POOLED"> <property name="driver" value="${jdbc.driverClassName}"/> <property name="url" value="${jdbc.url}"/> <property name="username" value="${jdbc.username}" /> <property name="password" value="${jdbc.password}" /> </dataSource> </environment> </environments>	<dataSource type="POOLED"> <property name="driver" value="${jdbc.driverClassName}"/> <property name="url" value="${jdbc.url}"/> <property name="username" value="${jdbc.username}" /> <property name="password" value="${jdbc.password}" /> </dataSource> </environment> </environments>

（二）Hibernate

数据库	KingbaseES	Mysql
方言包	定义 hibernate 配置文件，根据用户选择更改以下配置文件。 在 hibernate.properties 中增加如下声明： hibernate.dialect = org.hibernate.dialect.Kingbase8Dialect 在 hibernate.cfg.xml 中增加如下声明： <property name="dialect">org.hibernate.dialect.Kingbase8Dialect</property> 在 persistence.xml 中增加如下声明： <property name="hibernate.dialect" value="org.hibernate.dialect.Kingbase8Dialect" />	定义 hibernate 配置文件，根据用户选择更改以下配置文件。 在 hibernate.properties 中增加如下声明： hibernate.dialect = org.hibernate.dialect.MySQLDialect 在 hibernate.cfg.xml 中增加如下声明： <property name="dialect">org.hibernate.dialect.MySQLDialect</property> 在 persistence.xml 中增加如下声明： <property name="hibernate.dialect" value="org.hibernate.dialect.MySQLDialect" />
连接方式	<?xml version='1.0' encoding='UTF-8'?> <!DOCTYPE hibernate-configuration PUBLIC "-//Hibernate/Hibernate Configuration DTD 3.0//EN" "http://hibernate.sourceforge.net/hibernate-configuration-3.0.dtd">	<?xml version='1.0' encoding='UTF-8'?> <!DOCTYPE hibernate-configuration PUBLIC "-//Hibernate/Hibernate Configuration DTD 3.0//EN" "http://hibernate.sourceforge.net/hibernate-configuration-3.0.dtd"> <hibernate-configuration> <session-factory>

	<hibernate-configuration> <session-factory> <property name="hbm2ddl.auto">create-drop</property> <property name="show_sql">true</property> <property name="use_sql_comments">false</property> <property name="dialect">org.hibernate.dialect.Kingbase8Dialect</property> <property name="connection.driver_class">com.kingbase8.Driver</property> <property name="connection.url">jdbc:kingbase8://localhost:54321/test</property> <property name="connection.username">test</property> <property name="connection.password">test</property> <mapping resource="hibernate_test.hbm.xml" /> </session-factory> </hibernate-configuration>	<property name="hbm2ddl.auto">create-drop</property> <property name="show_sql">true</property> <property name="use_sql_comments">false</property> <property name="dialect">org.hibernate.dialect.MySQLDialect</property> <property name="connection.driver_class">com.mysql.jdbc.Driver</property> <property name="connection.url">jdbc:mysql://localhost:3306/test</property> <property name="connection.username">test</property> <property name="connection.password">test</property> <mapping resource="hibernate_test.hbm.xml"/> </session-factory> </hibernate-configuration>

（三）Django

数据库	KingbaseES	Mysql

连接方式	打开 settings.py 文件，然后修改 DATABASE 下的 default 键值进行修改即可。
	DATABASES = {
	'default': {
	'ENGINE':
	'django.db.backends.kingbase',
	'NAME': 'TEST',
		配置数据库（修改 settings.py 配置信息）
		DATABASES = {
		'default': {
		'ENGINE': 'django.db.backends.mysql',# 数据库引擎
		'NAME':'mydb',#数据库名字
		'USER':'root',#用户名
		'PASSWORD':'123456',#密码

	'USER': 'SYSTEM', 'PASSWORD': '123456', 'HOST': '127.0.0.1', 'PORT': '54321', }	'HOST':'127.0.0.1',# HOST 'PORT':'3306',#端口 'OPTIONS': {'charset':'utf8mb4'}, #打开数据库 编码格式 ——解决 4 字节表情无法储存问题 }
方言包	将 KingbaseES 方言包放入 Django 下的 dialects 文件夹中即可，例如 Django-4.0.6-for-py3	安装 Python 访问 MySQL 的模块 Django 官方已经不建议使用 pymysql 库了，而是改用 mysqlclient，直接 pip 安装即可。如： pip install mysqlclient

B.4 数据库对象、SQL 及 PLSQL

（一）数据类型转换

序号	MySQL 数据类型	KES 数据类型	功能说明	示例
1	ENUM(‘value1,value2’,...)	ENUM	枚举类型，枚举（enum）类型是由一个静态、值的有序集合构成的数据类型。	**Mysql:** 枚举类型 ENUM create table tt (name enum('y','n')); create table tt (name set('dianshi',' youxi')); **KingbaseES:** create type enumtype as enum ('y','n'); create table tt (name enumtype);
2	SET(‘value1’,value2’,...)	SET	集合数据类型：是可以具有 0 到 64 个值的自定义类型，每个值都必须在定义时指定的值列表中选取。指定包括多个 SET 成员的 SET 列值时各成员之间使用逗号（‘,’）分隔。	**Mysql:** create table myset (col SET('a','b','c','d')); **KingbaseES:** create type habbits as set(‘abc’,’def’,’ fff’); create table (c int,c2 habbits);

（二）操作符

序号	MySQL 操作符	KingbaseES 操作符	功能说明	示例
1	REGEXP RLIKE	REGEXP_LIKE（exp1,exp2）	正则匹配字符串。REGEXP 和 RLIKE 是 REGEXP_LIKE 同义词，KingbaseES 支持	**Mysql:** select 'Twet' regexp '^Tw.t$' AS 'Twet';

			REGEXP_LIKE，可以用 REGEXP_LIKE 函数改写，函数名和使用方式不同，返回值相同。	**KingbaseES:** select regexp_like('Abcde1','a[a-z]+');
2	|| 拼接字符串操作符	KingbaseES 需要使用拼接函数 concat	||拼接符和 concat 函数是将多个字段或字符串拼接为一个字符串。	Mysql: select 'hello' || ' world' KingbaseES: select concat('hello','world');

（三）有区别的函数

序号	MySQL 函数	KingbaseES 函数	功能说明	示例
数值函数				
1	TRUNCATE(X,D)	TRUNC(n1 [, n2])	按照小数位数，进行数值截取。函数名不同，同位置的函数参数含义相同，函数功能都是截取数字为指定的小数位数，返回 n1 截断到 n2 小数位，如 n2 省略，则将 n1 截断为 0 位，n2 可以为负数，以截断 n2 小数点左侧的数字。	**Mysql:** select truncate(123.4567, 3); # 123.456 select truncate(123.4567, 2); # 123.45 select truncate(123.4567, 1); # 123.4 select truncate(123.4567, 0); # 123 **KingbaseES:** select trunc(123.4567, 3); # 123.456 select trunc(123.4567, 2); # 123.45 select trunc(123.4567, 1); # 123.4 select trunc(123.4567, 0); # 123
字符函数				
1	REGEXP_SUBSTR(expr,pat[,pos[, occurrence[, match_type]]])	REGEXP_SUBSTR (source_char, pattern,[position,[occurrence,[match_parameter,[subexpr]]]])	正则匹配第 n 个字符在字符串中第 m 次出现的字符。 两个函数名相同，KingbaseES 中 regexp_substr （）返回正则表达式匹配的子字符串。支持六个参数，函数功能覆盖住 MySQL 的函数功能，6 个参数含义如下: source_char:源字符串 ； pattern:正则表达式; position:起始位置，默认为	**Mysql:** select regexp_substr('hello34 my 56 phone is 520 ', '[0-9]+',3,1) from dual; **KingbaseES:** select regexp_substr('hello34 my 56 phone is 520 ', '[0-9]+',3,1) from dual;

			1。表示从第几个字符串开始正则匹配; occurrence: 获取第几个正则匹配的分组值; match_parameter:一个字符串表达式，允许更改函数的默认匹配行为; subexpr: 指示函数将返回 pattern 中的哪个子表达式(0-9)，默认值为 0	

2	FORMAT(X,D[,locale])	round(numeric,s int)	将数字四舍五入到指定小数位。
			MySQL 中，函数将数字四舍五入到指定小数位。
			KingbaseES 中,ROUND 返回 n 四舍五入到 integer 小数点右边的位置。如果省略 integer，则将 n 四舍五入到零位。
				Mysql:
				select format(14500.2018, 2);
				
				KingbaseES:
				select round(14500.2088,'2');
				select round(14500.2088,2);

3	INSTR(str,substr)	INSTR(expr1 text,expr2 text,[expr3 int[,expr4 int]])	该函数表示第二个参数在第一个参数中的字符位置。 函数名一致，KingbaseES 的函数多了两个可选参数，如果在使用中不添加这两个可选参数， 函数功能和 MySQL 一致。 MySQL 中，第一个参数是字符串，第二个参数是要查找的字符串，返回第二个参数在第一个参数中出现的位置索引。 MySQL 中，返回子串在字符串中第一个匹配项的位置索引。 KingbaseES 中,扩展了位置参数。 第三个参数:此参数可选,如果省略默认为 1. 字符串索引从 1 开始。如果此参数为正，从左到右开始检索，如果此参数为负，从右到左检索，返回要查找的字符串在源字符串中的开始索引。	**Mysql:** select instr('helloworld','l') from dual; **KingbaseES:** select instr('helloworld','l') from dual; select instr('bcaaaaabbc', 'c',1,2);（mysql 不支持）

			第四个参数:代表要查找第几次出现的 string2. 此参数可选，如果省略，默认为 1.如果为负数系统会报错。	
4	LTRIM(str)	LTRIM(string text[,set text])	去除字符串左侧空格位。 函数名一致,KingbaseES 功能可以包含 MySQL 功能。MySQL 函数删除 str 字符串左侧的空格。KingbaseES 多一个可选参数 set，删除 set 中所有出现的字符，直到不存在 set 的字符,返回结果。Set 不指定，默认为空格。	**Mysql:** select ltrim (' hello world'); **KingbaseES:** select ltrim (' hello world'); select ltrim ('yxTomxx', 'xyz');

5	MAKE_SET(bits, str1,str2,...)	MAKE_SET(bits ,oid)	结果集返回。
			MySQL MAKE_SET()函数返回一个逗号分隔的字符串集合，该函数通过第一个参数对应的二进制决定是否其他字符串参数是否添加返回的结果集合中。
			函数名一致，参数不一致，MySQL 第一个参数是 bit，后面的参数是字符串，KingbaseES 第一个参数和 mysql 含义一致，第二参数是 set 集合类型的 oid 值。返回值一样。
			该函数返回一个 SET 值，该值由在 bits 中具有相应位的字符串组成。str1 对应位 0，str2 对应位。
			1，依次类推。str1,str2,…中的 NULL 值不会附加到结果中。与 mysql 相比，KingbaseES 中该值由指定 SET 类型在 bits 中具有相应位的字符串组成。
			bits = 1， 1 对应的二进制是 1，反过来是 1，那么 'a' 对应的 1，因此返回 'a'。
			bits = 2， 2 对应的二进制是 10,反过来是 01,参数 'a' 对应的 0，参数 'b' 对应的 1，因此返回 'b'。
				Mysql:
				select
				make_set(0, 'a', 'b', 'c', 'd'),
				make_set(1, 'a', 'b', 'c', 'd');
				
				KingbaseES:
				create type habbit2 as set('a','b','c','d');
				select * from sys_type where typname='habbit2';
				select make_set(1,16668);

			bits = 3， 3 对应的二进制是 11,反过来是 11,参数 'a' 对应的 1，参数 'b' 对应的 1，因此返回 'a,b'。 bits = 4， 4 对应的二进制是 100，反过来是 001，参数 'a' 对应的 0，参数 'b' 对应的 0，参数 'c' 对应的 1，因此返回 'c'	
6	RTRIM(str)	RTRIM(string text[,set text])	去除字符串右侧空格位。 函数名一致，函数删除指定字符串右侧的空格。KingbaseES 功能可以包含 MySQL 功能。MySQL 函数删除 str 字符串右侧的空格。KingbaseES 多一个可选参数 set，删除 set 中所有出现的字符，直到不存在 set 的字符，返回结果。Set 不指定，默认为空格。	**Mysql:** select rtrim ('hello world '); **KingbaseES:** select rtrim ('hello world '); select rtrim ('yxTomxx', 'xyz');

7	ELT(N,str1,str2,str3,...)	通过新建函数改写方式支持	返回字符串位置字符。
			在 MySQL 中，第一个参数是数字，表示要查找后面的第几个字符串，后面的参数均是字符串，主要功能是从后面的字符串列表中返回第 N 个字符串。
			Kingbase 可以通过新建函数方式改写
			create or replace function ELT(int, VARIADIC int[])
			returns int
			language plpgsql
			stable
			as
			fun
			begin
			return $2[$1];
			end;
			fun;
			类似这种改写方式，可以在实际应用中用到后，再针对性改写
				Mysql:
				select elt(1, '3','5');
				
				KingbaseES:
				select elt(1, '3','5');

8	INSERT(str,pos,len,newstr)	通过新建函数改写方式支持	替换指定字符函数。 在 MySQL 中，第一个参数是字	**Mysql:** select insert('Hello_World',

			符串，第二个参数是要替换的字符起始位置，第三个参数是从起始位置起要替换的字符的长度，第四个参数是要替换的字符串内容。
			KingbaseES 中可以通过新建函数方式改写
			CREATE or replace function insert(text, int, int, text)
			returns text
			LANGUAGE plpgsql
			immutable
			AS
			$$
			begin
			return substr($1, 1, $2 - 1) \|\| $4 \|\| substr($1, $2 + $3);
			end;
			$$;
			类似这种改写方式，可以在实际应用中用到后，再针对性改写
				6, 1, ' ');
				
				KingbaseES:
				select insert('Hello_World', 6, 1, ' ');

9	ifnull(expression, alt_value)	ifnull 函数在 KingbaseES 中参数需要使用单引号	IFNULL() 函数用于判断第一个表达式是否为 NULL,如果为 NULL 则返回第二个参数的值，如果不为 NULL 则返回第一个参数的值。	**Mysql:** SELECT IFNULL("Hello", "RUNOOB"); **KingbaseES:** SELECT IFNULL('Hello', 'RUNOOB');
时间日期函数				

1	NOW([fsp])	NOW()	返回当前时间。
			MySQL now 函数有可选参数，无参时候返回年月日时分表秒不带精度的日期时间，带参数时返回时区秒精度的系统时间。
			KingbaseES now 函数返回年月日时分表秒带 6 位精度的系统时间，不支持带参数 now 函数，不带精度时间可以用 substr 函数截取获得，或者使用 current_timestamp()函数
				MySQL:
				select now(6);
				select now();
				
				KingbaseES:
				select now();
				select substr(now(),1,18);
				select current_timestamp();

			替代。	

2	SUBDATE(date,INTERVAL expr unit),
	SUBDATE(expr,days)
		DATE_SUB(n,interval)	减去指定时间，返回新时间
			表示从指定的日期/时间(第一个参数）减去指定的时间间隔（第二个参数），并返回一个新的日期/时间。
			MySQL 函数支持 2 种参数类型，一种参数类型：第一个日期型，第二个参数是 interval 类型，也可以是整型表示天数。Kingbase 函数名是 DATE_SUB，（SUBDATE 是 DATE_SUB 函数的同义词）第一个参数是日期型，第二个参数是 interval 类型。
				Mysql:
				select subdate('2020-06-10',10);
				select subdate('2023-2-10','10 day');
				
				KingbaseES:
				select date_sub('2023-2-10','10 day');

（四）数据库对象操作

序号	MySQL 语句	KingbaseES 语句	功能说明	示例
1	表对象和列对象支持 comment	支持	表及列的注释内容 MySQL 支持在表和列创建 comment 以及通过 alter 方式修改 comment。 Kingbase 支持在表和列创建 comment 以及通过 alter 方式修改 comment，还可以通过单独的 comment 语句增加注释（MySQL 没有这种方式）	**Mysql:** create table oo (id int comment 'id'); alter table t1 modify column c1 varchar(20) comment '列的注释内容'; **KingbaseES:** create table oo (id int comment 'id'); alter table t1 modify column c1 varchar(20) comment '列的注释内容'; comment on column t1.c1 is 'c1 isname';
2	表对象创建时支持 collate	只支持数据库级的定义	表级 collate 定义 Kingbase 不支持，只支持库级别。	**Mysql:** create table `table1` (`id` bigint(20) unsigned not null auto_increment, `field1` text not null comment '字段 1',`field2` varchar(128) not

				null default '' comment '字段2', primary key (`id`))engine=innodb collate=utf8_unicode_ci; **KingbaseES:** create database tt lc_collate 'sv_SE.utf8';
3	分区表支持range、list、hash分区及子分区	支持range、list、hash分区及子分区	MySQL和Kingbase都支持range、list、hash分区及子分区，创建语法不同。 MySQL还支持KEY、LINEAR HASH和LINEAR KEY分区，KES不支持LINEAR HASH和LINEAR KEY这两类分区。	**Mysql:** 范围分区： create table staff(id int(32) not null, code_ varchar(30), fname varchar(30), time_ date, primary key(`id`,`time_`)) partition by range(id)(partition p0 values less than (5), partition p1 values less than (10), partition p2 values less than (15), partition p3 values less than (MAXVALUE)) 列表分区： CREATE TABLE employees (id INT NOT NULL, fname VARCHAR(30), lname VARCHAR(30), hired DATE NOT NULL DEFAULT '1970-01-01', separated DATE NOT NULL DEFAULT '9999-12-31', job_code INT, store_id INT) PARTITION BY LIST(store_id) (PARTITION pNorth VALUES IN (3,5,6,9,17), PARTITION pEast VALUES IN (1,2,10,11,19,20), PARTITION pWest VALUES IN (4,12,13,14,18), PARTITION pCentral VALUES IN (7,8,15,16)); 哈希分区： create table staff(id int(32) not null, code_ varchar(30), fname varchar(30), time_ varchar(30), PRIMARY key(`id`,`time_`))

				partition by hash(id) partitions 3; **KingbaseES:** 范围分区： CREATE TABLE pkslow_person_r (age int not null, city varchar not null) PARTITION BY RANGE (age); create table pkslow_person_r1 partition of pkslow_person_r for values from (MINVALUE) to (10); create table pkslow_person_r2 partition of pkslow_person_r for values from (11) to (20); create table pkslow_person_r3 partition of pkslow_person_r for values from (21) to (30); create table pkslow_person_r4 partition of pkslow_person_r for values from (31) to (MAXVALUE); 列表分区： - 创建主表 create table pkslow_person_l (age int not null, city varchar not null) partition by list (city); -- 创建分区表 CREATE TABLE pkslow_person_l1 PARTITION OF pkslow_person_l FOR VALUES IN ('GZ'); CREATE TABLE pkslow_person_l2 PARTITION OF pkslow_person_l FOR VALUES IN ('BJ'); CREATE TABLE pkslow_person_l3 PARTITION OF pkslow_person_l DEFAULT; 哈希分区： -- 创建主表 create table pkslow_person_h (age int not null,

				city varchar not null) partition by hash (city); -- 创建分区表 create table pkslow_person_h1 partition of pkslow_person_h for values with (modulus 4, remainder 0); create table pkslow_person_h2 partition of pkslow_person_h for values with (modulus 4, remainder 1); create table pkslow_person_h3 partition of pkslow_person_h for values with (modulus 4, remainder 2); create table pkslow_person_h4 partition of pkslow_person_h for values with (modulus 4, remainder 3);
4	数据库中执行数据导出 SELECT... INTO OUTFILE	改写支持	KingbaseES 输出到文件，Kingbase 通过 copy to 来实现。	**Mysql:** vi /etc/my.cnf secure-file-priv = /home/mysql service mysqld restart select * from t1 into outfile "/home/mysql/fact_sale_20210526.txt"; **KingbaseES:** Copy t1 to '/usr1/proj/sql/tabledata'
5	数据库中执行数据导入 source sql 文件的绝对路径	改写支持	KingbaseES 支持从文件中读取数据存储到表中，可以通过copy from来实现，示例： Copy t2 from '/usr1/proj/sql/tabledata' Copy 语句支持转义、分隔符、换行符、指定数据格式、编码、跳过最大错误行等参数，具体使用请参考 KingbaseES SQL 参考手册	**Mysql:** source /home/mysql/test2.sql **KingbaseES:** Copy to t2 from '/home/kingbase/t2.sql';

6	RENAME	改写支持	KingbaseES 不支持直接的 rename 语句，但是各个对象的 alter 语句可以支持更改对象名称，示例：Alter table t1 rename to newt1;	**Mysql**: rename table t2 to t22; **KingbaseES:** alter table t2 rename to t22;
7	UPDATE...table_references WHERE 多表更新	改写支持	KingbaseES 支持	**Mysql:** update product p INNER JOIN product_price pp ON p.productid= pp.productid SET pp.price = p.price * 0.8; **KingbaseES:** create table aa (id int,name varchar(10)); create table bb (id int,name varchar(10)); update aa set aa.id=bb.id,aa.name=bb.name from bb where aa.id=bb.id;
8	INSERT INTO...ON DUPLICATE KEY UPDATE...VALUES(colname)	支持 Mysql 写法，并且也支持 on conflict 写法	MySQL 中更新子句中可以引用被插入的列值。	**Mysql:** create table `user2` (`id` int(11) not null auto_increment, `username` varchar(94) not null, `age` int(11) default null, `gender` int(1) default null, `type` int(1) default null, primary key (`id`), unique key `idex_name` (`username`) using btree, key `idx_type` (`type`) using btree) engine=innodb auto_increment=13 default charset=utf8; insert into user2(username, age, gender) values("saint33", 99, 1) on duplicate key update age = 88; **KingbaseES:** create table t2 (id int primary key,name varchar(20)); Insert into t2 values(1,'自来水

				'); insert into t2 values(1,'可乐') on conflict (id) do update set name=excluded.name;

（五）plsql 操作

序号	MySQL 操作	KingbaseES 操作	功能说明	示例

1	声明变量需添加语句 DECLARE	存储过程或者函数创建过程中，直接声明变量，无需加 declare 关键字	用于在 plsq 程序中声明变量。	**Mysql:**
				create procedure p8 ()
				begin
				declare a int;
				declare b int;
				set a=5;
				set b=5;
				insert into t values (a);
				select s1 * a from t where s1>= b;
				end;
				KingbaseES:
				\ set SQLTERM /
				create procedure p8 () as
				a int ;
				b int ;
				begin
				insert into t values (a);
				select s1 * a from t where s1>= b;
				end;
				/

2	Set 赋值	变量名:= values 方式	plsql 赋值方式
			Mysql 为 set a = a+1
			KingbaseES 改为 a:=a+1
				Mysql:
				CREATE PROCEDURE p8 ()
				BEGIN
				DECLARE a INT;
				DECLARE b INT;
				SET a=5;
				SET b=5;
				INSERT INTO t VALUES (a);
				SELECT s1 * a FROM t WHERE s1>= b;
				END;
				
				KingbaseES:
				create or replace function test(a int)
				returns int as $$

				declare
				  a int;
				begin
				  a:=0;
				  a:=a+1;
				  raise notice 'aaa:%',a;
				  return a;
				end;
				$$
				LANGUAGE plsql;

3	DELIMITER //	\set SQLTERM /	多行输入方式设置	**Mysql:**
				SET GLOBAL
				log_bin_trust_function_creators = 1;
				delimiter//
				create function testsum(num int)
				returns int
				begin
				declare i int default 0;
				declare sum int default 0;
				lable1:loop
				  set sum=sum+i;
				  set i=i+1;
				  if i>num then
				  leave lable1;
				  end if;
				end loop lable1;
				return sum;
				end//
				KingbaseES:
				\set SQLTERM /
				CREATE PROCEDURE dorepeat(p1 INT)
				AS
				X INTEGER :=0;
				BEGIN
				WHILE x <= p1 LOOP
				x := x + 1;
				END LOOP;
				RAISE notice 'x=%' , to_char(x);
				END
				/

4	结果输出方式，	结果输出方式	结果输出方式，	**Mysql:**

	SELECT count;	RAISE notice 'hello' ;	具体使用参见下方示例	create procedure pro_test1()
				begin
				select 'Hello Mysql' ;
				end
				call pro_test1();
				KingbaseES:
				create procedure pro_test1() as
				begin
				raise notice 'Hello Mysql' ;
				end

5	ITERATE label	改写支持	ITERATE 语句表示再次启动 loop 循环
			LEAVE 语句表示退出标记的循环语句,KingbaseES 不支持该语句，但是可以通过 loop 结合 continue（再次启动循环）语句改写达到语句的目的。
				Mysql:
				CREATE PROCEDURE myProc ()
				BEGIN
				DECLARE count INT default 0;
				increment: LOOP
				SET count = count + 1;
				IF count = 5 THEN ITERATE increment; END IF;
				SELECT count;
				IF count > 10 THEN LEAVE increment;
				END IF;
				END LOOP increment;
				END
				call myProc();
				
				KingbaseES:
				\set SQLTERM /
				CREATE PROCEDURE myProc ()
				as
				i INT := 0;
				BEGIN
				LOOP
				i = i + 1;
				CONTINUE WHEN i = 5;
				raise notice 'i 的值为：%',i;
				EXIT WHEN i > 10;
				END LOOP;
				END;
				call myProc()

6	LEAVE label	改写支持		
7	REPEAT 语句	改写方式支持	重复执行语句，直到满足条件退出 Kingbase 不支持该语句，但是可	**Mysql:** create function dorepeat(p1 int) returns int

			以通过 while loop 语句改写。	BEGIN
				SET @x = 0;
				REPEAT
				SET @x = @x + 1;
				UNTIL @x > p1 END REPEAT;
				return @x;
				END
				KingbaseES:
				CREATE PROCEDURE dorepeat(p1 INT)
				AS
				X INTEGER :=0;
				BEGIN
				WHILE x <= p1 LOOP
				x := x + 1;
				END LOOP;
				RAISE notice 'x=%' , to_char(x);
				END

8	WHILE 语句
	WHILE search_condition DO statement_list END WHILE
		WHILE boolexpr LOOP Statement END LOOP	WHILE 循环语句，语法规则 MySQL 和金仓不同，具体使用示例参见上例。	**Mysql:**
				SET GLOBAL log_bin_trust_function_creators = create procedure test_while (IN in_count INT)
				BEGIN
				declare count INT default 0;
				while count < 10 do
				set count = count + 1;
				end while;
				select count;
				END
				
				KingbaseES:
				CREATE PROCEDURE dorepeat(p1 INT)
				AS
				X INTEGER :=0;
				BEGIN
				WHILE x <= p1 LOOP
				x := x + 1;
				END LOOP;
				RAISE notice 'x=%' , to_char(x);
				END

9	游标声明 DECLARE cursor_name	游标变量声明： type cursorname is REF CURSOR	两者的声明方式不同，MySQL 中使用游标需要先声明游标，再打开	**Mysql:** CREATE TABLE test1(a int,b int); INSERT INTO test1 VALUES

	CURSOR FOR SELECT * FROM table_name; 打开游标 OPEN cursor_name; 关闭游标 CLOSE cursor_name;	RETURN *%ROWTYPE CURSOR cursor is selectstatement;	游标，最后关闭游标,kingbaseES 可以不需要手动打开和关闭游标	(1,2),(3,4),(5,6); CREATE TABLE test2(a int); INSERT INTO test2 VALUES (100),(200),(300); CREATE TABLE test3(b int); INSERT INTO test3 VALUES (400),(500),(600); CREATE TABLE test3(b int); INSERT INTO test3 VALUES (400),(500),(600); select fun1(100) /*创建函数*/ CREATE FUNCTION fun1(v_max_a int) RETURNS int BEGIN /*用于保存结果*/ DECLARE v_total int DEFAULT 0; /*创建一个变量，用来保存当前行中 a 的值*/ DECLARE v_a int DEFAULT 0; /*创建一个变量，用来保存当前行中 b 的值*/ DECLARE v_b int DEFAULT 0; /*创建游标结束标志变量*/ DECLARE v_done int DEFAULT FALSE; /*创建游标*/ DECLARE cur_test1 CURSOR FOR SELECT a,b from test1 where a<=v_max_a; /*设置游标结束时 v_done 的值为 true，可以 v_done 来判断游标是否结束了*/ DECLARE CONTINUE HANDLER FOR NOT FOUND SET v_done=TRUE; /*设置 v_total 初始值*/ SET v_total = 0; /*打开游标*/ OPEN cur_test1; /*使用 Loop 循环遍历游标*/ a:LOOP /*先获取当前行的数据，然后将当

				前行的数据放入 v_a,v_b 中，如果当前行无数据，v_done 会被置 为 true*/
				FETCH cur_test1 INTO v_a, v_b;
				/*通过 v_done 来判断游标是否结束了，退出循环*/
				if v_done THEN
				LEAVE a;
				END IF;
				/*对 v_total 值累加处理*/
				SET v_total = v_total + v_a + v_b;
				END LOOP;
				/*关闭游标*/
				CLOSE cur_test1;
				/*返回结果*/
				RETURN v_total;
				END
				/*结束符置为;*/
				KingbaseES:
				1、示例 1
				CREATE OR REPLACE FUNCTION test()
				RETURNS numeric AS
				$BODY$
				--
				--测试显式游标和隐式游标的效率
				--
				declare
				vamt numeric(14,2);
				cur cursor is select * from wk_piece3;
				rec record;
				begin
				--1)显示游标
				/*
				vamt := 0;
				open cur;
				loop
				fetch cur into rec;
				exit when not found;
				vamt := vamt + rec.amount;
				end loop;
				close cur;
				*/

				--2）隐式游标
				vamt :=0;
				for rec in select * from wk_piece3 loop
				vamt := vamt + rec.amount;
				end loop;
				return vamt;
				end;$BODY$
				LANGUAGE plsql VOLATILE
				2、示例 2：
				Declare
				CURSOR c1 RETURN STUDENT%ROWTYPE;
				---声明方式 1
				CURSOR c2 is
				SELECT ID, name,score from student WHERE score>90;
				CURSOR c1 RETURN STUDENT%ROWTYPE is
				SELECT * from student where id=1;
				BEGIN
				NULL;
				END;

10	DECLARE {EXIT|CONTINU} HANDLER FOR {error-number |SQLSTATE error-string| condition}
	#####
	关键词解释:
	CONTINUE：表示继续执行当前 SQL 脚本。
	EXIT：表示终止执行当前 SQL 脚本。
		改写方式支持	KingbaseES 不支持这样的错误处理语句，有自己的错误处理语句 exception_name EXCEPTION;	**Mysql:**
				1、示例 1
				create table test1(id int primary key);
				insert into test1 values(99);
				CREATE PROCEDURE test1()
				BEGIN
				DECLARE CONTINUE HANDLER FOR SQLSTATE '23000' SELECT '主键冲突 Duplicate key';
				insert into test.test1 SELECT 99;
				insert into test.test1 select 88;
				END;
				/
				
				2、示例 2
				CREATE PROCEDURE test1()
				BEGIN

				DECLARE EXIT HANDLER FOR SQLSTATE '23000' SELECT '主键冲突 Duplicate key';
				insert into test.test1 SELECT 99;
				insert into test.test1 select 88;
				END;
				/
				
				KingbaseES:
				create table tt01(name varchar(20) primary key);
				delete from tt01;
				select * from tt01;
				
				declare
				begin
				insert into tt01 values (1);
				insert into tt01 values (1);
				insert into tt01 values ('a');
				exception
				when others then
				RAISE EXCEPTION 'mwh 设置的错误输出：(%)', SQLERRM;
				end;

附 录 C
（资料性）
关系型数据库国产化迁移手册示例

C.1 引言

为了提高数据库的性能和可扩展性，同时满足信息科技创新要求，我们计划将现有的 Mysql 数据迁移到国产数据库金仓数据库中。本文档将详细介绍迁移的流程、操作步骤、预期成果及风险应对等方面内容。

C.2 目标

C.2.1 迁移现有 Mysql 数据到新的金仓数据库。
C.2.2 提高数据库性能和可扩展性。
C.2.3 确保迁移过程中应用服务的高可用性。

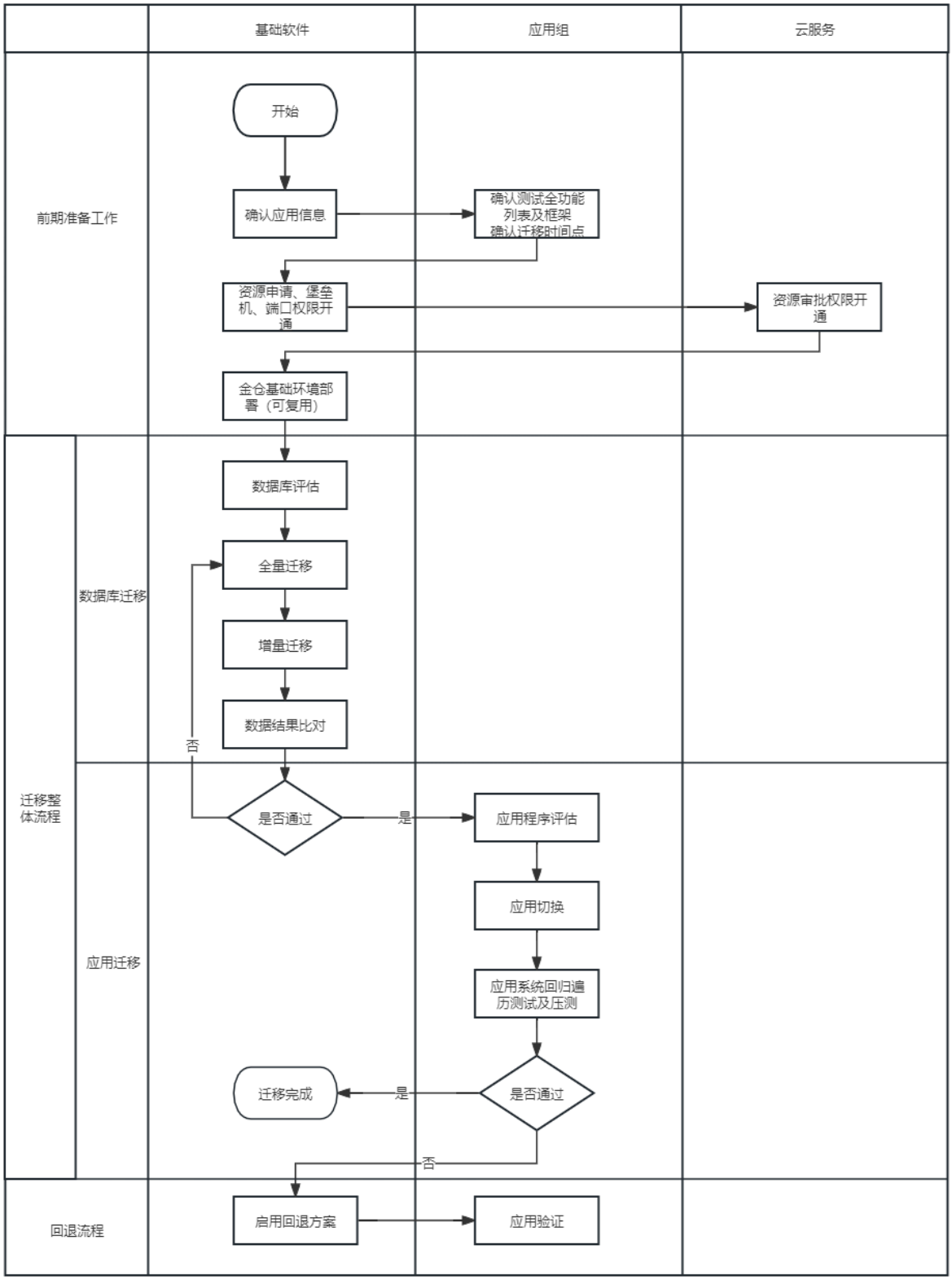
C.3 迁移规划

C.3.1 总体迁移流程

分类	目录			编号	前置项	人力资源
	一级	二级	三级			组别
前期准备工作	基础设施部署	应用背景收集及基础设申请	确认待迁移应用信息，包括应用数据量、版本,迁移方式（全量/增量）、mysql 权限梳理、连接方式等相关信息	1.1.1		基础软件
			应用确认测试全功能列表及框架	1.1.2		应用组
			确认应用迁移替换时间点	1.1.3		应用组
			数据库堡垒机、VPN、硬件资源申请，网络访问端口权限开通（数据库工具和应用到数据库）	1.1.4		基础软件
		金仓基础环境部署	部署环境检查	1.2.1	1.1	基础软件
			金仓数据库环境部署	1.2.2	1.1	基础软件
			KFS 部署-可复用现有环境	1.2.3	1.1	基础软件
			迁移工具 KDTS 部署-可复用现有环境	1.2.4	1.1	基础软件
			KDMS 部署-可复用现有环境	1.2.5	1.1	基础软件
迁移整体流程	数据库用迁移	工程评估	KDMS 结构采集	2.1.1	1.2	基础软件
			采集结构包	2.1.2	1.2	基础软件
			KDMS 迁移评估	2.1.3	1.2	基础软件
			迁移评估报告	2.1.4	1.2	基础软件
		结构迁移	KDMS 翻译转换	2.2.1	2.1	基础软件
			结构迁移-执行转换后脚本	2.2.2	2.1	基础软件
		工作通知	不兼容语法，需要代码工作	2.3.1	2.2.2	

		数据迁移	KDTS 离线全量迁移	2.4.1	3.2.1、2.3.1	基础软件
			KFS 在线增量同步	2.4.2	2.4.1	基础软件
		结果比对	KFS 结构&数据	2.5.1	2.4.2	基础软件
	应用迁移	**应用程序**	配置应用 SQL 采集 agent 信息（与应用部署在一起）	3.1.1	2	基础软件
			启动应用 SQL 接收及存储服务	3.1.2	3.1.1	基础软件
			启动应用	3.1.3	3.1.2	基础软件
			点击应用业务功能，进行 SQL 采集和存储	3.1.4	3.1.3	基础软件
			采集数据打包	3.1.5	3.1.4	基础软件
			上传 KDMS 进行评估和改写	3.1.6	3.1.5	基础软件
			KDMS 动态 SQL 采集	3.1.7	3.1.6	基础软件
			KDMS 动态 SQL 迁移评估	3.1.8	3.1.7	基础软件
			迁移评估报告	3.1.9	3.1.8	基础软件
	应用迁移	**应用迁移和适配**	提前 1 天应用程序封版	3.1.1	2.5.1	应用组
			进行连调配置，切换 mysql 到金仓	3.1.2	3.1.1	应用组
			应用系统回归遍历测试及压测	3.1.3	3.1.2	应用组
			应用程序适配改造	3.1.4	3.1.3	应用组
		工作确认	应用改造是否成功	3.2.1	3.1.4	应用组
			切换后代码封版（一周）	3.2.2	3.2.1	应用组
回退流程	**数据库回切**	**数据库回切**	KFS 数据库回退（金仓到 mysql）	4.1.1		基础软件
	应用回切	**应用回切**	进行连调配置，切换（金仓到 mysql）	4.2.1	4.1.1	应用组

迁移过程中大部分时间不影响原有业务正常运行，只有最后应用切换验证的时候会存在短暂影响。迁移流程示意图如下：



C.3.2 前期准备阶段（基础软件&应用组）

C.3.2.1 硬件服务器资源准备，申请开通堡垒机资源权限及源数据库端口访问权限。

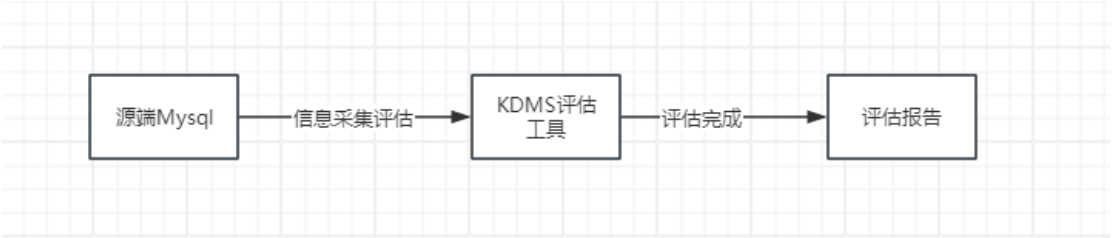
C.3.2.2 数据库环境以及迁移工具部署。

C.3.2.3 应用确认测试全功能列表及架构、应用迁移替换时间点。

C.3.3 数据库迁移（基础软件）

C.3.3.1 数据库评估

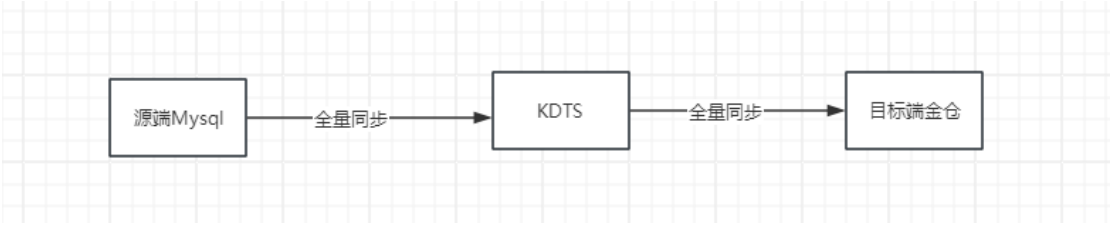
在评估过程中，KDMS 评估工具会根据预设的评估指标和算法，生成一份详尽的评估报告。这份报告将包含数据库迁移的可行性分析、迁移所需的时间和资源预估、潜在风险的解决方案以及优化建议等内容。



C.3.3.2 全量迁移

通过 KDTS 迁移工具将源端 mysql 数据平滑迁移至目标端金仓数据库中。

全量迁移流程如下：

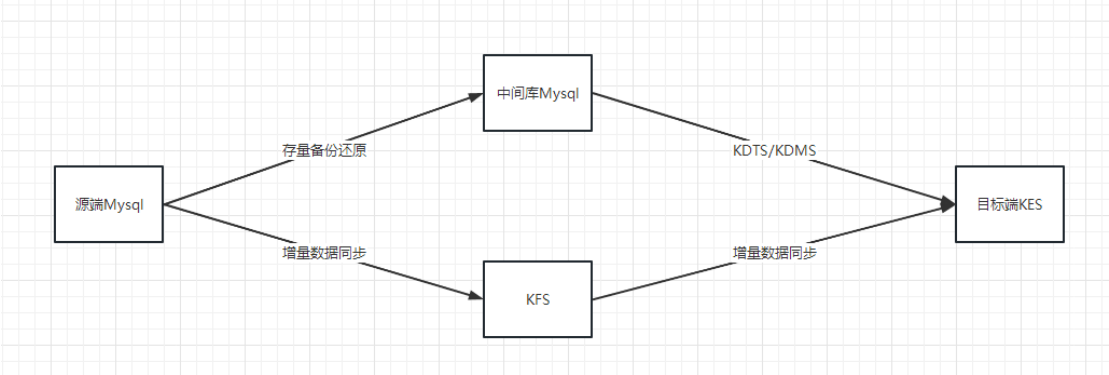


配置文件准备：

```
sources:
  #源数据库类型
  - dbType: MySQL
    #源数据库版本(5.5,5.7,8.0,8.0.29),小于5.5版本的5.x统一使用5.5
    dbVersion: 8.0
    #连接字符串，格式为：jdbc:mysql://{host}:{port}/{database}
    #MySQL连接串必须添加参数zeroDateTimeBehavior=convertToNull&nullCatalogMeansCurrent=true&yearIsDateType=true
    #并根据MySQL数据库的字符集添加与之匹配的字符集参数，例如：useUnicode=true&characterEncoding=utf-8
    url: jdbc:mysql://127.0.0.1:3306/polaris?zeroDateTimeBehavior=convertToNull&nullCatalogMeansCurrent=true&yearIsDateType=true&tinyInt1isBit=false&autoReconnect=true&failOverReadOnly=false&autoReconnectForPools=true&allowPublicKeyRetrieval=true&useSSL=false&useUnicode=true&characterEncoding=utf-8&serverTimezone=GMT%2B8 #&socketTimeout=3600000
    #驱动类名
    driverClassName: com.mysql.cj.jdbc.Driver #5.6版本及以上
    #driverClassName: com.mysql.jdbc.Driver #5.5版本及以下
    #用户名
    username: root
    #用户名是否使用密文（使用bin/cryptotool.sh或cryptotool.bat生成）
    ciphertextUsername: false
    #密码
    password: Faw-MySql8_One_2021
    #密码是否使用密文（使用bin/cryptotool.sh或cryptotool.bat生成）
    ciphertextPassword: false
    #要迁移的模式，多个模式用英文逗号隔开，如：A,B,C（不要回车换行），也可填入"*"（需要双引号，将迁移源库中此用户所能访问的所有模式）
    #schemas: SCHEMA1,SCHEMA2,SCHEMA3
    #schemas: "*"
    schemas: build_center,collaboration_center,beacon,dev_center,qingyun,document_center,alita,efficiency_center,dayu_auth,project_center,dayu_login_account,fip-admin,learn_center,polaris
    #不迁移的模式，多个模式用英文逗号隔开，如：A,B,C（不要回车换行）
    schemaExcludes:
    #数据库连接验证SQL
    connectionTestSql: select 1
    #获取连接后是否进行测试
    validateOnGetConnection: true
    #验证连接的有效性
```

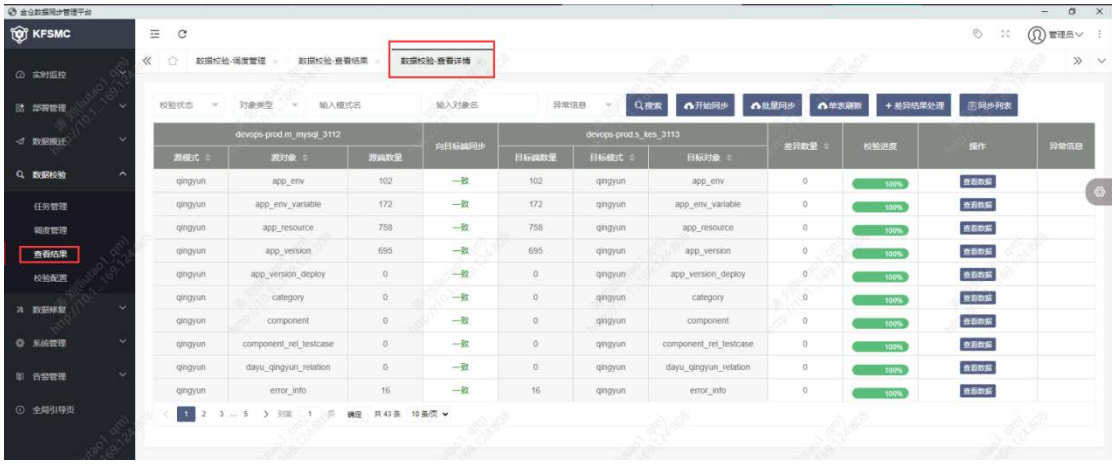
C.3.3.3 增量迁移

增量迁移是在全量迁移的基础上进行补充，通过中间库以及 kfs 同步工具实现增量数据更新。Kfs 同步工具设置双向策略，先开启 Mysql 到金仓数据库链路，等应用测试验证无误后，再打开从金仓数据库到 Mysql 的链路。



C.3.3.4 数据结果比对

通过数据结果比对来验证源数据库与目标数据库是否存在差异。



C.3.4 应用迁移和适配验证（应用组）

C.3.4.1 开发人员进行配置修改，实现数据库连接以及适配工作。
C.3.4.2 启动应用，验证是否能够正常访问新的数据库。
C.3.4.3 对迁移后的系统进行全面测试，确保功能的正确性和性能的稳定性。
C.3.4.4 对比迁移前后的系统性能指标，评估迁移的效果。

C.3.5 生产上线阶段（基础软件&应用组）

C.3.5.1 建议对源 Mysql 数据库进行全量备份。
C.3.5.2 启用生产环境中的与金仓数据库连接的应用。
C.3.5.3 申请域名绑定等相关信息。
C.3.5.4 监控系统运行状态，及时处理可能出现的问题。
C.3.5.5 对迁移过程进行总结和优化，为后续项目提供经验。

C.4 预期成果和影响

C.4.1 提高数据库性能和可扩展性，支持公司业务的快速发展。
C.4.2 支持高并发访问、持久化存储能力，满足多样化的应用需求。
C.4.3 减少数据库维护成本，提高数据安全性与可靠性。
C.4.4 对业务几乎没有影响，保证业务连续性和高可用性。

C.5 风险评估与应对策略

C.5.1 系统性能风险：迁移过程可能导致系统性能下降。

应对策略：选择在业务低峰期进行迁移，并准备相应的应急预案，如回滚操作等。

C.5.2 迁移失败风险：迁移过程中或迁移后发现存在重大问题需要进行数据的回退。

应对策略：迁移过程中发现问题停止迁移即可，若在迁移后运行一段时间后进行回退，因为 kfs 同步工具开启了双向同步策略，即源端和目标端数据互相同步，应用将数据库连接地址改回源 Mysql 进行验证即可。

C.6 总结

附录 C 介绍了 Mysql 迁移到金仓数据库的相关操作，包括迁移的目标、计划、预期成果和影响以及可能存在的风险和应对策略。在实施迁移前，我们需要充分评估各种风险和影响，并制定相应的应对策略以确保整个过程的顺利进行。