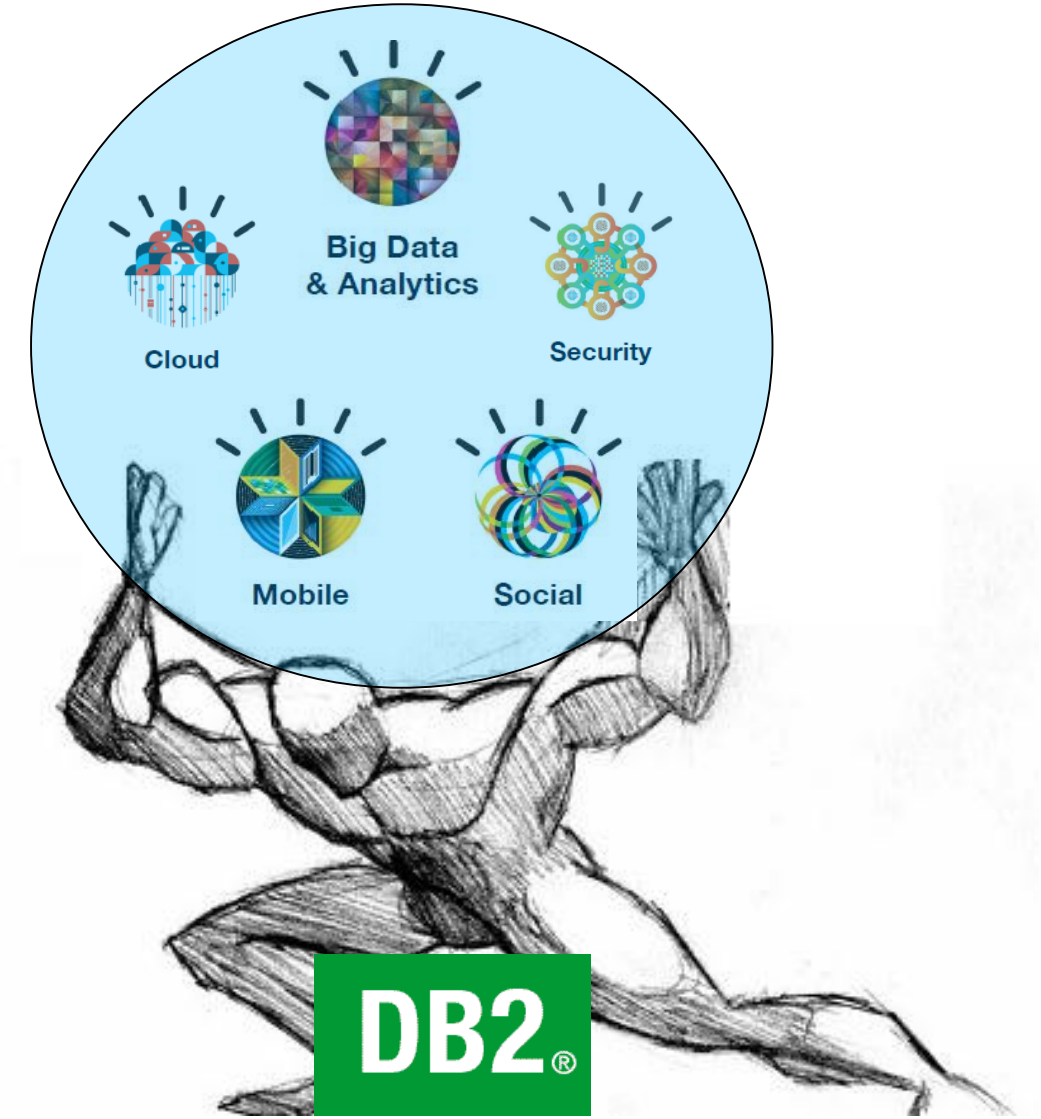
Very Large Database (VLDB) and DB2 for i

Power Systems 2015

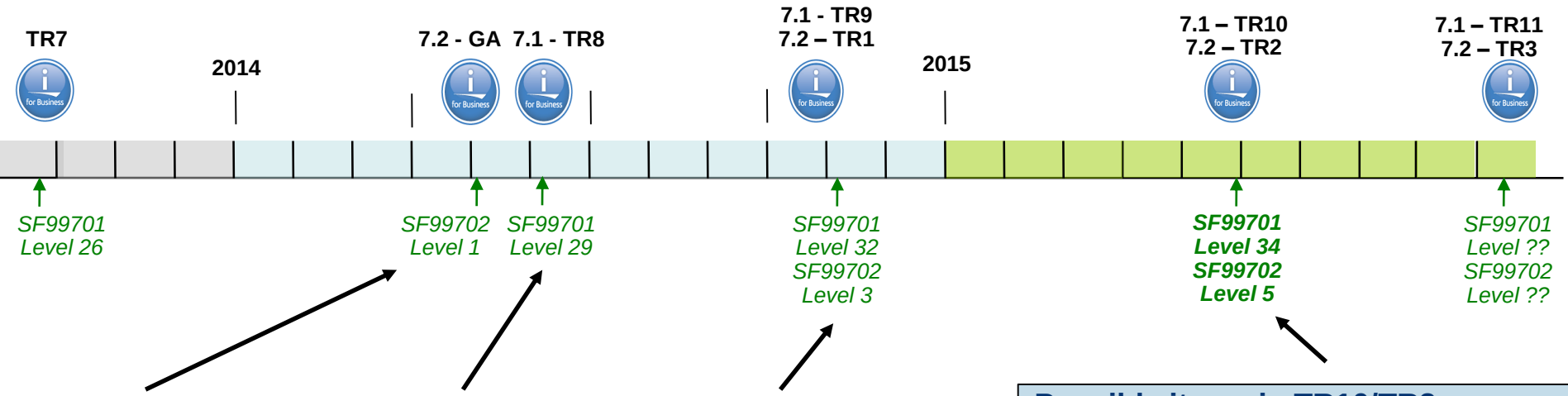
Scott Forstie
forstie@us.ibm.com

DB2 for i 业务架构师
DB2 for i Business Architect



DB2 for i – DB2 PTF Groups 提供的增强

IBM i 7.1 & 7.2



7.2 - GA

- Row & Column Access Control (RCAC)
- All IBM i 7.1 Enhancements thru TR8

7.1 - TR8&TR9 timed Enhancements:

- Generate SQL procedure
- Program & Package statement level statistical catalogs
- Pipelined Functions
- Regular Expressions
- Padding Scalar Functions

7.2 – TR1 timed Enhancements:

- CREATE **OR REPLACE** support on MASK and PERMISSION
- DB2 Built-in Global Variables for job name
- System Limits Phase 3 - IFS
- And more…

Possible items in TR10/TR2:

- Create OR REPLACE table
- JSON DB2 Store Technology Preview
- SQE Performance improvements
- Limits to growth extension for SQL functions
- More DB2 for i Services
- New SQL built-in functions
- Enhancements for Guardium users
- Enhancements for SAP on i clients
- And more…

记录通过PTF所提供的增强功能的文档:

www.ibm.com/developerworks/ibmi/techupdates/db2

Plans are subject to change without notice until formally announced

DB2 for i

Big Data & Analytics

- Limits to Growth 增长的极限值
- Very Large Database 超大数据库

持续的投资与革新

V5R3
- Partitioned tables
- Online Reorganize

V5R4
- SSD Memory Preference
- On Demand Performance Center
- Health Center
- 2M SQL statement
- 1000 tables in a query

IBM i 6.1
- DECFLOAT
- Grouping sets / super groups
- Expression in Indexes
- Deferred Restore of MQT and Logicals
- Environmental limits
- Program Type SUB
- DB2 Statistical catalogs

IBM i 7.1
- XML Support
- EVI INCLUDE support for grouping set
- 1.7 Terabyte Indexes
- Teraspace Storage Model
- Index Advice for OR predicates
- System Limits for DB2 tables
- SQL Plan Cache controls

IBM i 7.2
- Increased Timestamp precision within SQL tables
- Multiple-action Triggers
- System Limits for IFS
- SQL support for Memory preference
- Automatic repositioning of updated rows within a partitioned table

IBM i 7.next
- **Enhancements to DB2 for i**

2004 2006 2008 2010 2014 2016

极限值设计

Database Constructs	FIPS	Oracle 12c	Microsoft SQL Server 2012	DB2 for i 7.2	DB2 for LUW V10.5	DB2 for z/OS V11
Length of an identifier (in bytes)	18	30	128	128	128	128
Length of CHARACTER data type (in bytes)	240	2000	8000	32766	254	255
Decimal precision of NUMERIC data type	15	38	38	63	31	31
Decimal precision of DECIMAL data type	15	38	38	63	31	31
Decimal precision of INTEGER data type	9	38	10	10	10	10
Decimal precision of SMALLINT data type	4	38	5	5	5	5
Binary precision of FLOAT data type	20	126	53	53	53	53
Binary precision of REAL data type	20	63	24	24	24	21 or 24
Binary precision of DOUBLE PRECISION data type	30	126	53	53	53	53
Columns in a table	100	1000	1024	8000	1012	750
Values in an INSERT statement	100	1000	1024	8000	1012	750
Set clauses in an UPDATE statement	20	1000	1024	8000	1012	750
Length of a row (not including LOBs)	2000	3996739	8060	32766	32677	32714
Columns in a UNIQUE constraint	6	32	16	120	64	64
Length of a UNIQUE constraint	120	.5 DB_BLOCK_SIZE	900	32K	8192	2000
Length of foreign key column list	120	.5 DB_BLOCK_SIZE	900	32K	8192	2000
Columns in a GROUP BY clause	6	255	Limited by length	Limited by length	1012	750
Length of GROUP BY column list	120	DB_BLOCK_SIZE	8060	32766	32677	16000
Sort specifications in ORDER BY clause	6	255	Limited by length	Limited by length	1012	750
Length of ORDER BY column list	120	DB_BLOCK_SIZE	8060	32766	32677	16000
Columns in a referential integrity constraint	6	32	16	120	64	64
Tables referenced in a SQL statement	15	No limit	No limit	1000	No limit	225
Cursors simultaneously open	10	OPEN_CURSORS	2 gig	No limit	No limit	No limit
Items in a select list	100	1000	4096	8000	1012	750

Largest limit

VLDB – Very Large Database 超大数据库

How large is "big data" on IBM i?

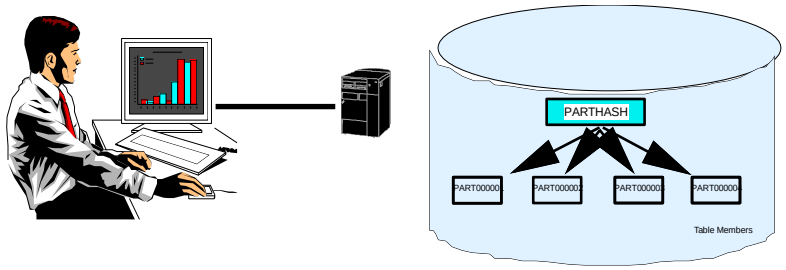
Non-partitioned table 非分区表	**1.7 terabytes**
Local Partitioned table 本地分区表	**435.2 terabytes**
Non-partitioned index 非分区索引	**1.7 terabytes**
Local Partitioned index 本地分区索引	**435.2 terabytes**

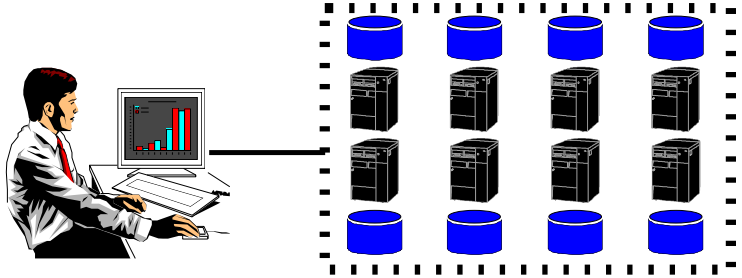
Local Partitioning 本地数据分区

需要付费选项激活

- **DB2 Symmetric Multiprocessing (57nnSS1 Boss option 26)**
- **DB2 Multisystem (57nnSS1 Boss option 27)**
- **HA Journal Performance (57nnSS1 Boss option 42)**
- **Query Manager and SQL Development Kit (57nnST1)**

DB2 Multisystem – Boss Option 27





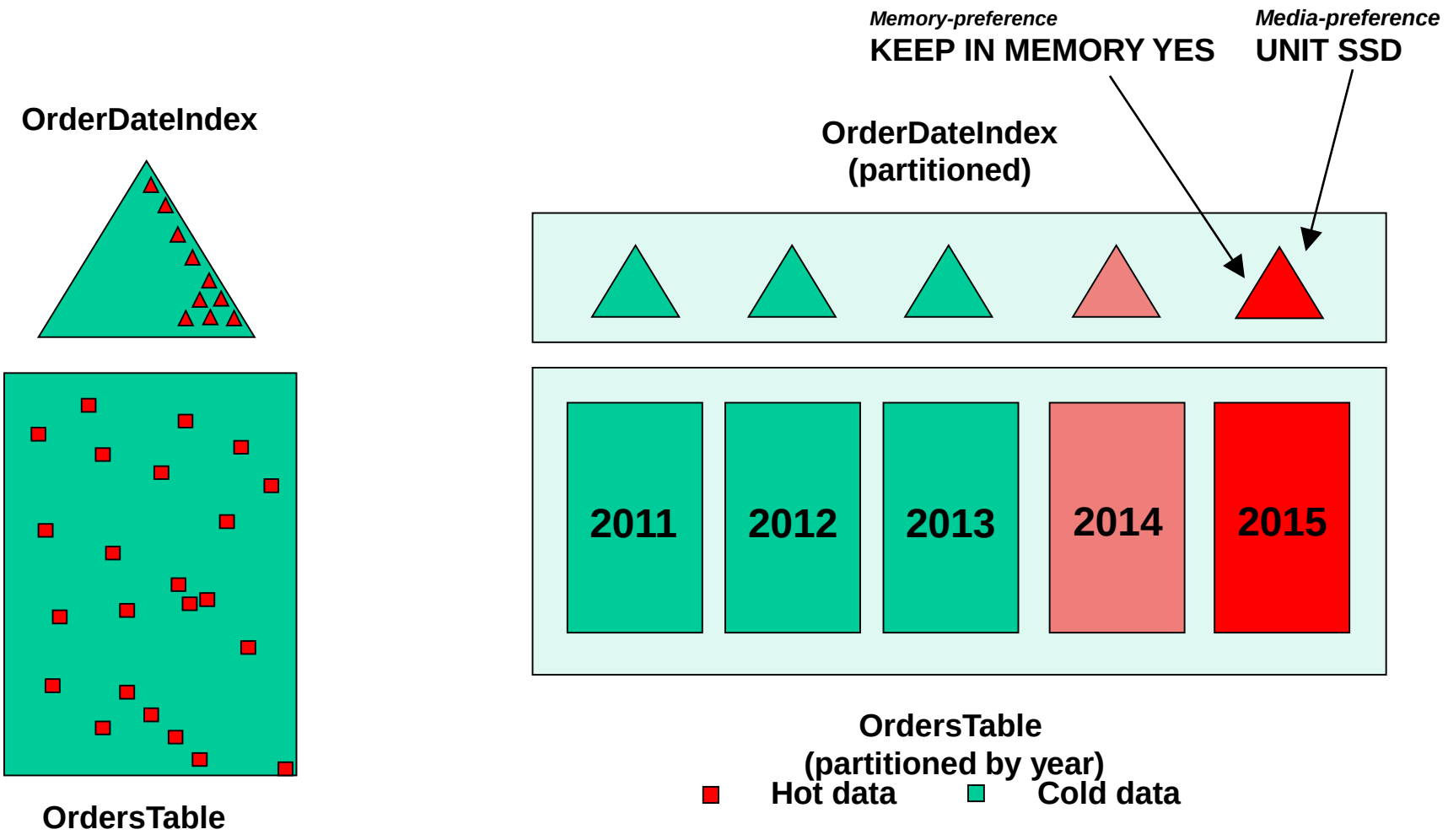
- **Local Partitioned tables**
  本地分区表
  - **分区对SQL应用透明**
  - **哈希散列值分区与区间分区**
  - **数据分布于各分区 (members)**
  - **单表最多可扩展至256个分区**

常用推荐

- Distributed partitioned tables
  分布式分区表
  - 分区对所有应用透明
  - 哈希值分区 (用户可控制对应哈希值的存放位置)
  - 数据分布于个IBM i 节点 (系统或逻辑分区)
  - 单表最多可扩展到32个系统上

一般不用

DB2 for i 可扩展性 – 示例



数据分区 – 概念

- 分区是物理文件的成员(**MEMBERS)**
- 两种数据分区
  - **Hash** (**由数据库决定**如何安排行)
  - **Range** (**由用户决定**如何安排行)
- 可通过SQL Alias(别名)、视图或 override(覆盖) 来引用单独某个分区
- SQL 视整个表为一个大的成员.
  原生访问仍如以前一样仅能访问某一个单独的成员.
- SQL Query Engine (SQE) 内嵌避免访问不必要的分区的技术，如果有 WHERE 子句构建于分区键字上

Partitioning Scalar Functions 分区标量函数

DATAPARTITIONNAME	返回行所在的分区名
DATAPARTITIONNUM	返回行所在的分区号
DBPARTITIONNAME	返回行所在的数据库名
DBPARTITIONNUM	返回行所在的节点号
HASH	返回一组值的分区号
HASHED_VALUE	返回行对应的分区映射索引号

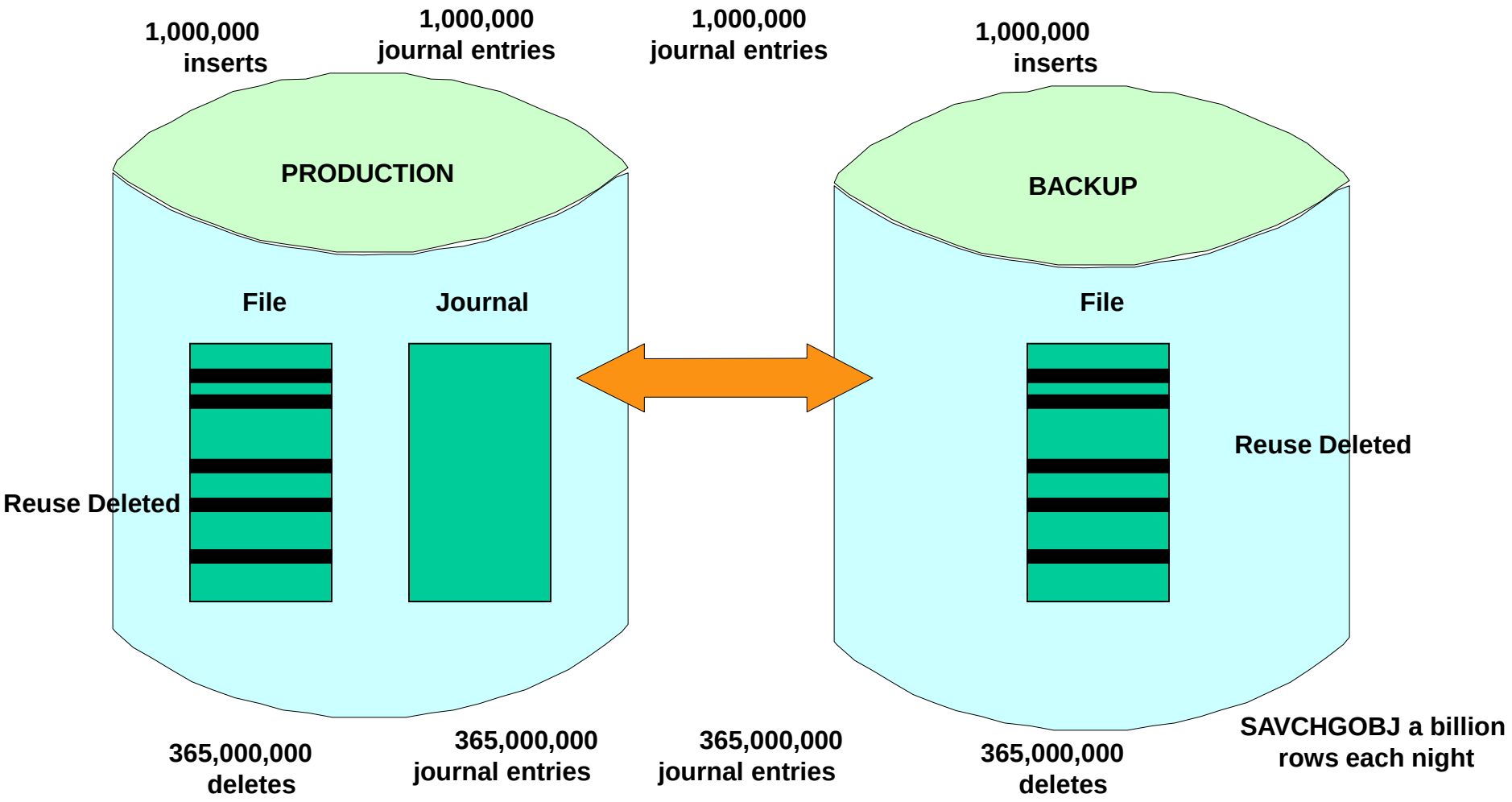
数据分区的好处

	好处
性能	▪ 查询可以**高效**的仅针对指定分区 ▪ **策略性**使用高速存储设备
管理	▪ **快速**转入和转出 ▪ 如果所有**改动的数据**仅涉及部分分区，可加快备份/恢复
高可靠性 & 灾难恢复	▪ 对使用逻辑复制的场景，可**减少**日志传输量

数据分区的好处 – 示例

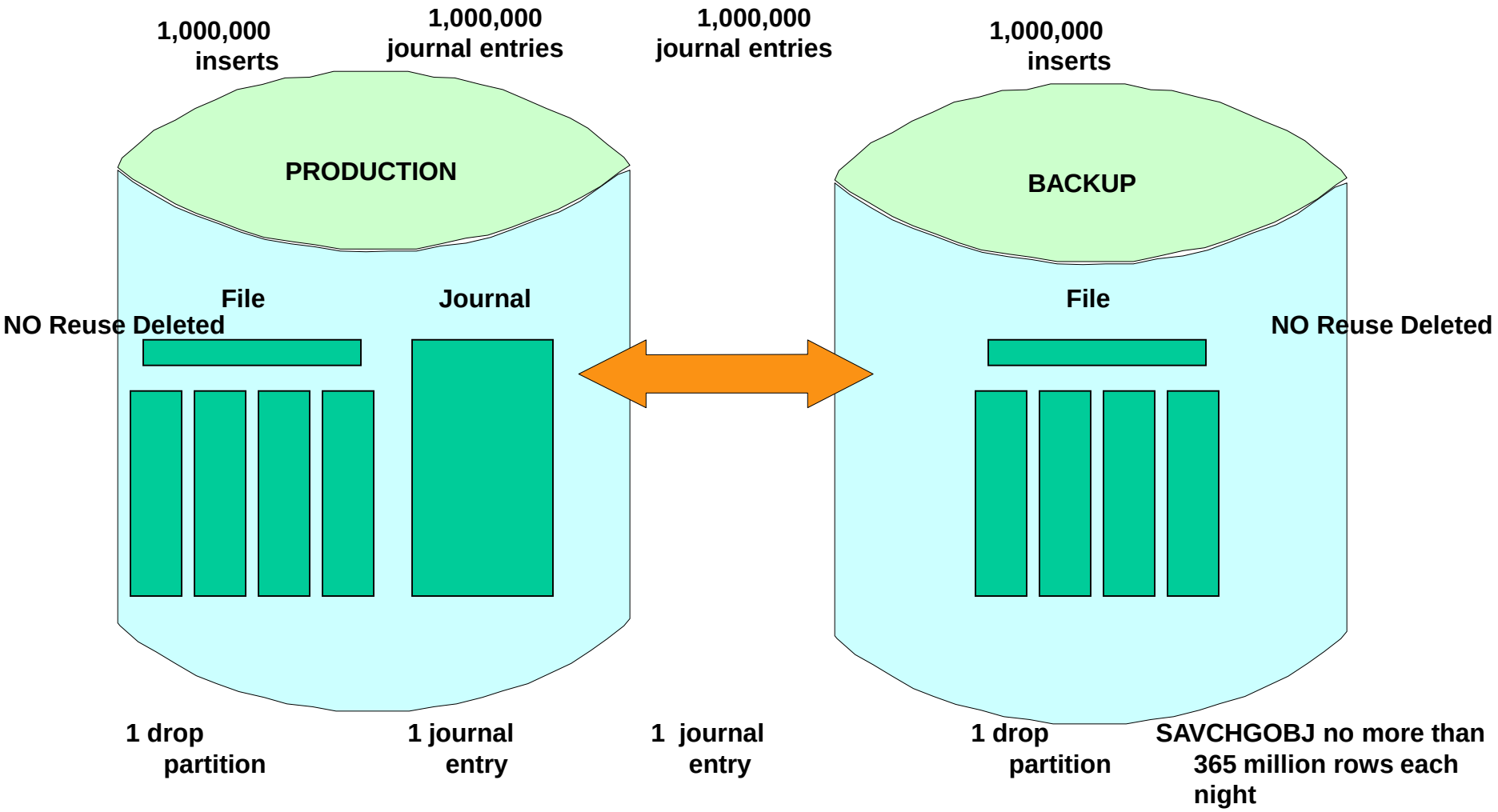
	业务流程
假设场景	▪ 一个交易文件每天有**1百万条数据插入** ▪ 表包含**日期和时间戳**的列 ▪ 生产环境的主表要求保存**三年的数据** ▪ 每开始新的一年，最陈旧的一年的**数据会被清除** ▪ **生产库每天使用Save while active方式备份** ▪ HA使用**逻辑复制**至备份系统

非数据分区示例



每年年终清理最早一年数据时，会发生什么情况…

数据分区示例



删除最早一年数据变得非常简单高效

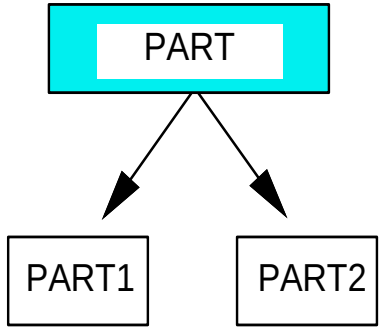
数据分区的缺点

- 性能下降的**可能性**
  - 通常，分区表中分区数据越大，SQL数据修改和数据操作的额外代价越大。用户在满足需求的情况下应该设计尽可能小的分区数来最小化这些额外的代价。
  - 当访问分区数据时，强烈建议使用大于1的并行度(SMP)。
- 原生数据库访问可能需要使用逻辑文件和记录格式选择程序(FMTSLR)来通过逻辑文件执行写入操作。
- **从非数据分区 → 数据分区的转型**

创建分区表

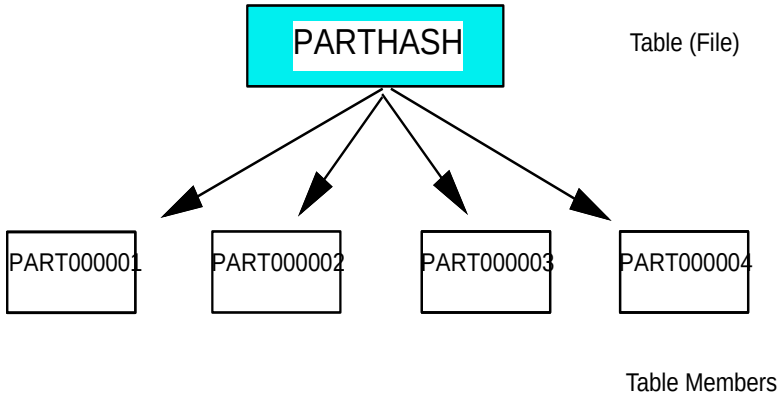
RANGE

```
CREATE TABLE part (
   c1 INTEGER DEFAULT NULL ,
   c2 CHAR(10) CCSID 37 DEFAULT NULL )
   PARTITION BY RANGE ( c1 NULLS LAST )  (

   PARTITION  part1
   STARTING ( MINVALUE ) INCLUSIVE
   ENDING ( 100 ) INCLUSIVE ,

   PARTITION  part2
   STARTING ( 101 ) INCLUSIVE
   ENDING ( MAXVALUE ) INCLUSIVE )
```



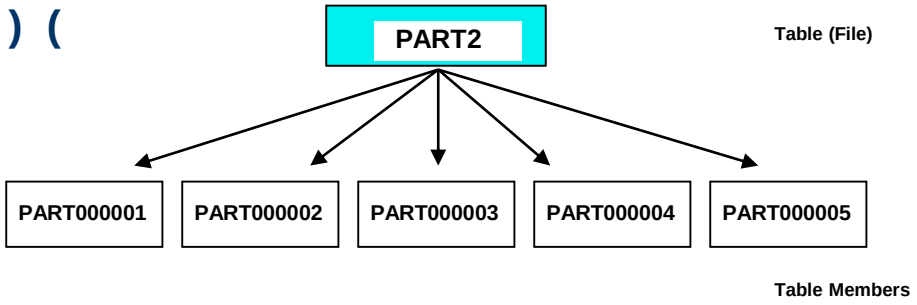
HASH

```
CREATE TABLE parthash (
    c1 INTEGER DEFAULT NULL ,
    c2 CHAR(10) CCSID 37 DEFAULT NULL )
    PARTITION BY HASH (c1)
    INTO 4 PARTITIONS
```



创建分区表

```
CREATE TABLE part2 (
   c1 DATE DEFAULT NULL ,
   c2 CHAR(10) CCSID 37 DEFAULT NULL )

   PARTITION BY RANGE ( c1 NULLS LAST ) (

   STARTING ( '2014-01-01' ) INCLUSIVE
   ENDING     ( '2015-01-01' ) EXCLUSIVE
   EVERY      ( 1 YEAR )   )
```



- 创建多个分区的另一种写法
- 本示例中，每个分区包含一年数据

创建数据分区

- **分区化的计划**
  - 确认理解数据是如何被分布的
  - 分析数据应该如何在各分区中分布
  - 和 Lab Based Services 一起创建 VLDB 策略

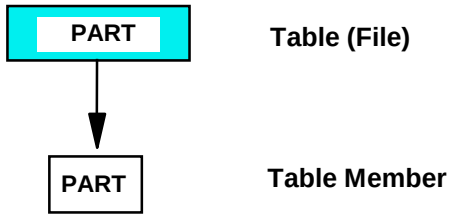
- **使用分区表的限制**
  - 分区范围不能互相重叠
  - Floating point, Decimal Floating point, LOB, DataLink 和 ROWID，这些数据类型不能作为分区键字
  - 不能超过256个分区

- **已经被取消的分区表限制**
  - 标识列不能在分区表中使用
  - 引用约束不能在分区表中使用
  - 分区表不能作为引用约束的父表
  - 当一行被更新时，不能自动移到另一个分区

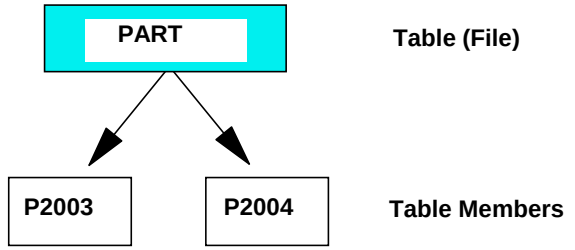
IBM i 7.1中取消的限制
IBM i 7.2中取消的限制

更改分区表

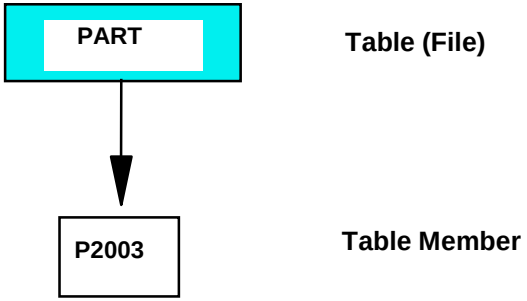
```
CREATE TABLE newpart (
   c1 DATE DEFAULT NULL ,
   c2 CHAR(10) CCSID 37 DEFAULT NULL )
```




```
ALTER TABLE newpart
   ADD PARTITION BY RANGE ( c1 NULLS LAST )
   (
   PARTITION  p2003
   STARTING ( MINVALUE )  INCLUSIVE
   ENDING     ( '2013-01-01' )  INCLUSIVE ,
   PARTITION  p2004
   STARTING ( '2014-01-01' )  INCLUSIVE
   ENDING     ( MAXVALUE ) INCLUSIVE )
```



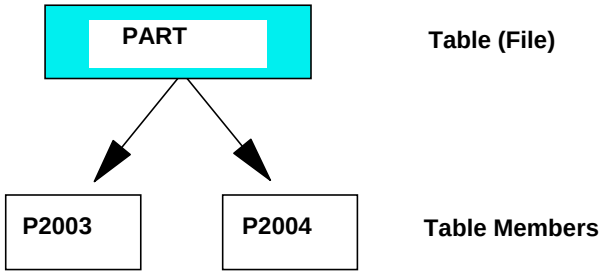

```
ALTER TABLE newpart
   DROP PARTITIONING
```



更改分区表

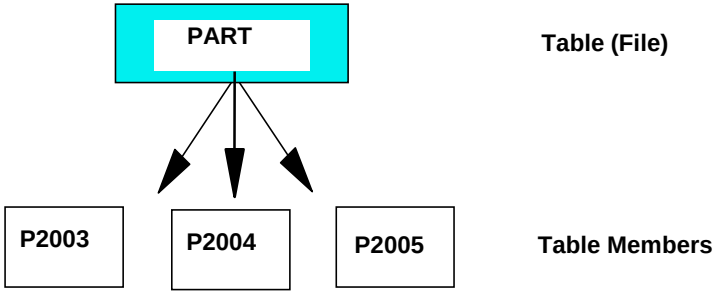
ALTER TABLE newpart
ALTER PARTITION p2004

STARTING (‘2004-01-01’) INCLUSIVE
ENDING (‘2005-01-01’) EXCLUSIVE

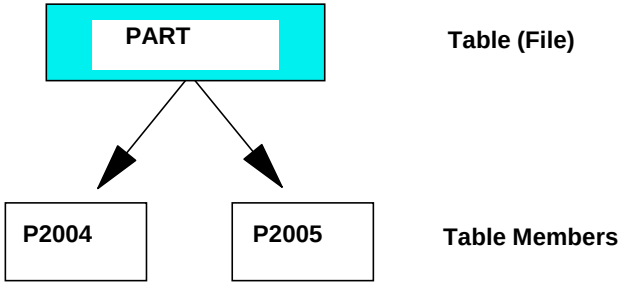


ALTER TABLE newpart
ADD PARTITION p2005

STARTING (‘2005-01-01’) INCLUSIVE
ENDING (MAXVALUE) INCLUSIVE



ALTER TABLE newpart
DROP PARTITION p2003
DELETE ROWS



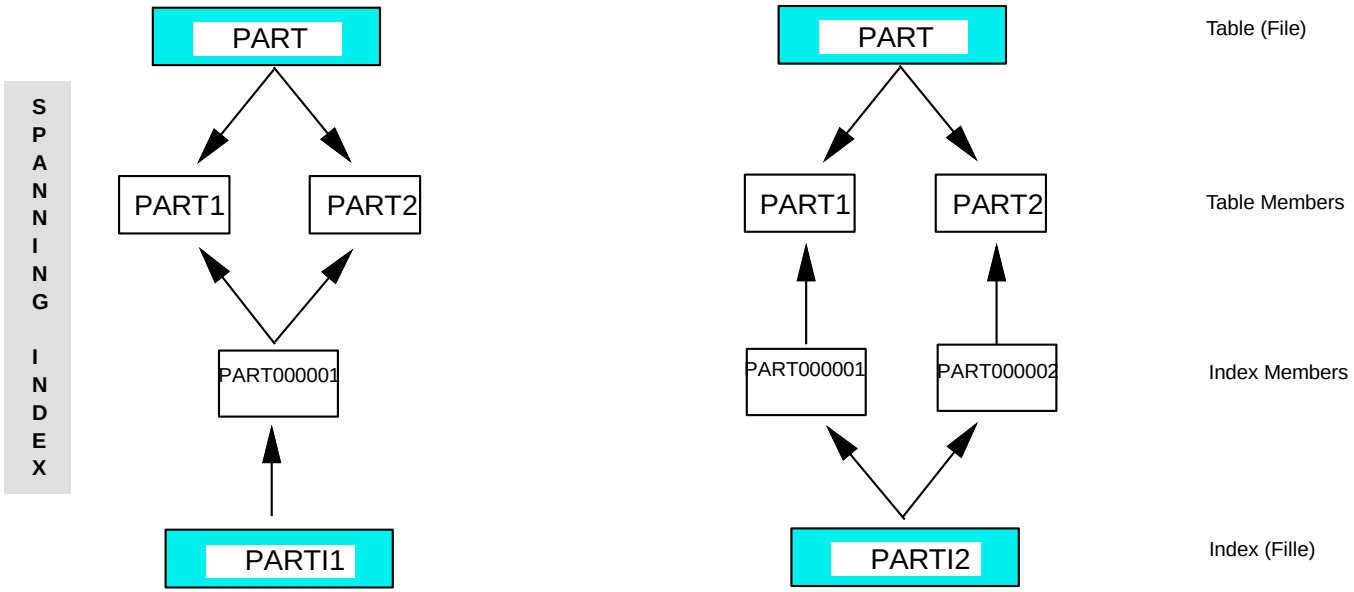
更改分区表

- **计划** 任何修改分区的尝试之前
  - 确认理解数据如何分布

```
WITH x AS ( SELECT  CASE WHEN c1 < '2013-01-01' THEN 'p2003'
                         WHEN c1 < '2014-01-01' THEN 'p2004'
                                                ELSE 'p2015'
                    END AS partname FROM newpart)
SELECT partname AS "Part Name", COUNT(*) AS "Count"
FROM x
GROUP BY partname
```

- **将操作打包在一个单独的 ALTER 语句中来减少 数据的移动**
- **不要低估改变分区所需的时间**
  - 增加或删除一个分区是非常快的，除非有非分区的索引
  - 更改分区将导致数据移动及索引重建
  - 从非分区表更改至分区表将导致数据移动及索引重建，反之亦然
- **限制**
  - **物理文件不能更改成分区表。您必须先转化至SQL表。**
  - DROP COLUMN 对于分区键字列是不允许的
  - 表中所有数据必须在指定的分区范围内

分区索引

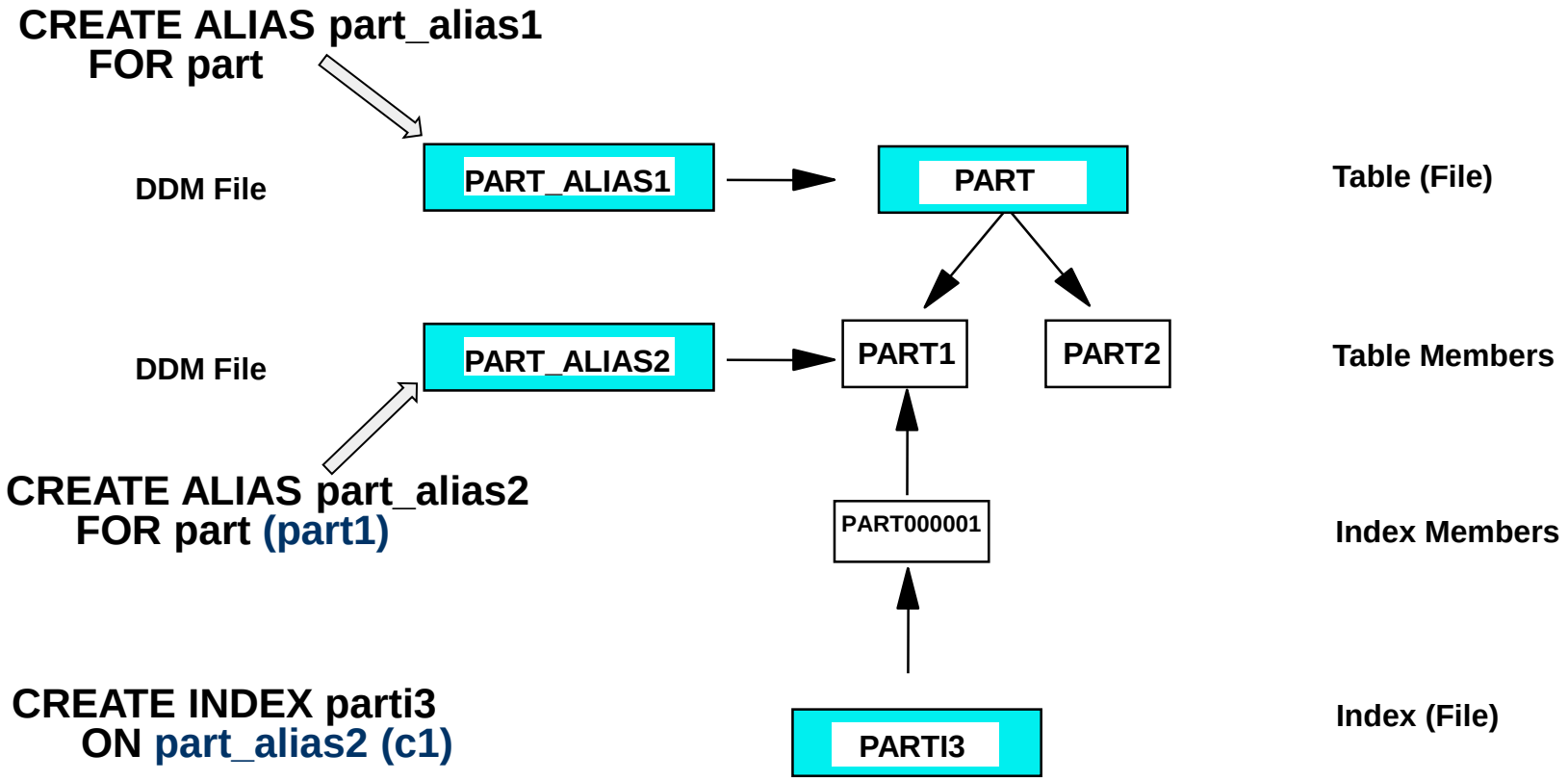


CREATE INDEX parti1
ON part (c1)
NOT PARTITIONED

CREATE INDEX parti2
ON part (c1)
PARTITIONED

- **限制**
  - **如果一个分区的 索引是唯一的， 索引的键字必须和分区键字相同或是分区键字的父集。**
    否则，唯一索引必须是非分区索引。

Aliases 别名

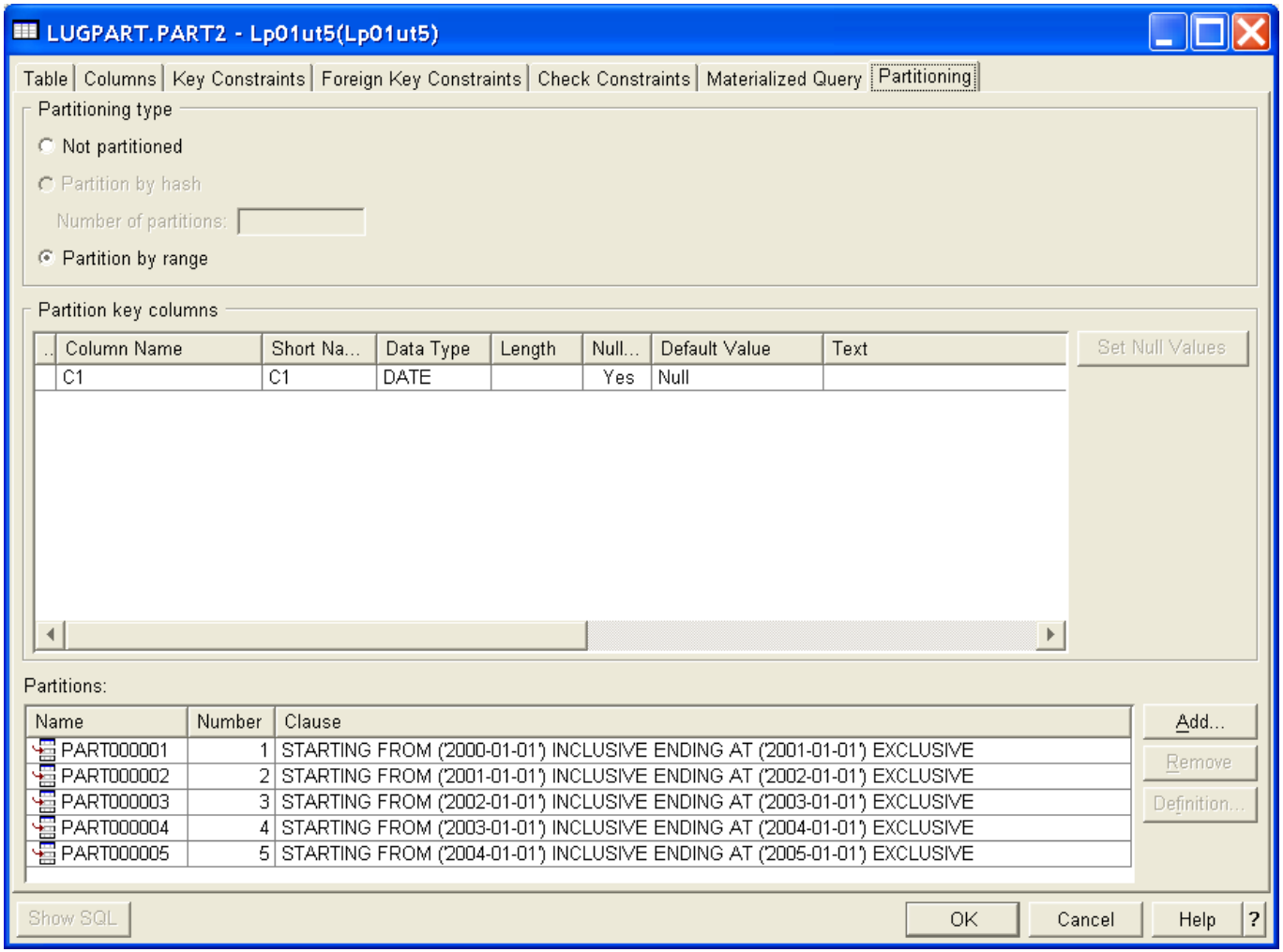


- 限制
  - 别名不能在 SQL schema 相关语句中使用.

目录（Catalogs）& 分区

Catalog Name	Purpose
SYSPARTITIONDISK	分区使用的硬盘
SYSPARTITIONMQTS	基于分区创建的物化视图表（MQT）
SYSPARITITIONSTAT	表分区的统计信息及属性
SYSPARTITIONINDEXDISK	表分区上的索引的硬盘信息
SYSPARTITIONINDEXES	表分区上的索引
SYSPARTITIONINDEXSTAT	表分区上的索引的统计信息

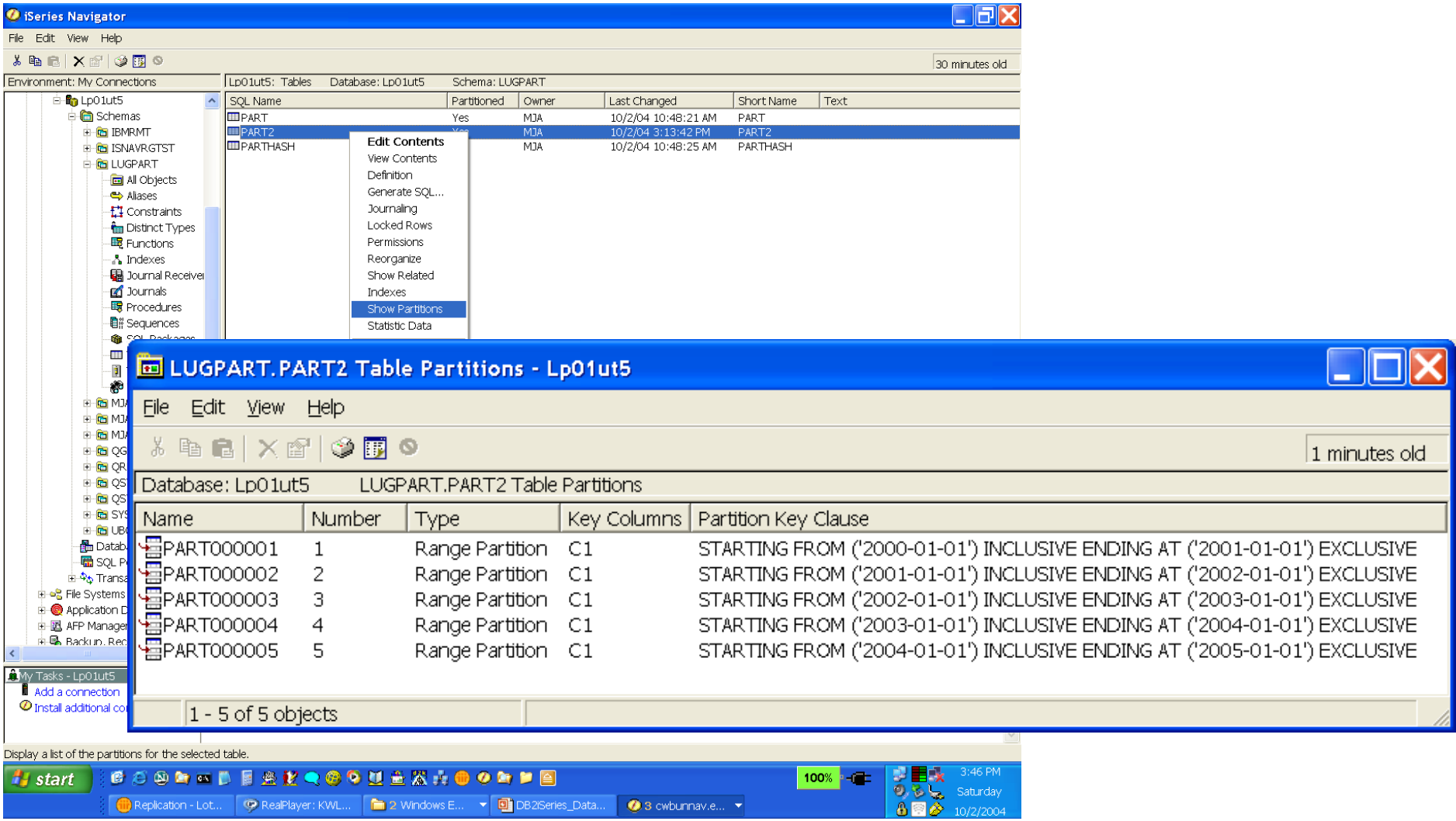
Navigator for i – 表分区定义



在建表向导 中同样支持

iSeries Navigator

Navigator for i – 显示分区



数据分区 – 信息资源

- **White Paper:**
  Table Partitioning Strategies for DB2 for i
  https://ibm.biz/PartitionedTablesIBMi

- **Mike Cain Blog thread:**
  http://db2fori.blogspot.com/2013/03/living-large.html

- **DB2 for i - VLDB Consulting Workshop:**
  https://ibm.biz/DB2CoEworkshops

Backup

Futures – Items under consideration

- Alter Enhancements
  - Add partition while active
  - Add a partition without requiring DDS spanning indexes built with *ALL to be recreated
  - Asynchronously move data
  - Attach and detach partitions
- Additional partitioning methods (size, etc.)
- Increase maximum number of partitions

These enhancement ideas are under consideration