Physical Design: DB2

Physical Structures

- Primary structure:
  - Heap
  - Array (Range-clustered tables)
- Indexes:
  - dense B+-trees

- Indexes are bidirectional by default: they allow forward and reverse scans

Range-clustered tables

- Array primary structure
- The table should have an integer key that is tightly clustered (dense) over the range of possible values.
- The columns of this integer key must not be nullable, and the key should logically be the primary key of the table.
- The allocation of all the space for the complete set of rows in the defined key sequence range is done during table creation,

Secondary indexes

- Secondary indexes contain only keys and record IDs in the index structure.
- The record IDs always point to rows in the data pages.
- Dense indexes

CREATE TABLE

CREATE TABLE [schema.] table (column_definition [table_constraint] [,...*n*])

[physical_properties]

[partitioning-clause]

column_definition, table_constraint

- As in SQL Server
- PRIMARY KEY constraint:
  - A unique index will automatically be created for the columns
  - The name of the index will be the same as the constraint-name

UNIQUE constraint

- A unique secondary index will automatically be created for the columns
- The name of the index will be the same as the constraint-name

[physical_properties]

[ORGANIZE BY KEY SEQUENCE sequence-key-spec]

[IN tablespace]

ORGANIZE BY KEY SEQUENCE

```
ORGANIZE BY KEY SEQUENCE
    (column-name
    [STARTING FROM constant]
    ENDING AT constant
    [...n])
```

- Defines a range-clustered tables
  - The data type of the column must be SMALLINT, INTEGER, or BIGINT
  - STARTING and ENDING specify the range

IN tablespace

- Defines the tablespace where the table is created

[partitioning-clause]

```
CREATE TABLE ACCESSNUMBERS
(AREA INTEGER, EXCHANGE INTEGER)
PARTITION BY RANGE (AREA, EXCHANGE)
(
STARTING (1,1) ENDING (10,100),
STARTING (11,1) ENDING (MAXVALUE,MAXVALUE)
)
```

- Two partitions

[partitioning-clause]

DISTRIBUTE BY HASH (column-name,…)

- Specifies the use of the default hashing function on the specified columns, called a *distribution key*, as the distribution method across database partitions.

```
CREATE TABLE SALES
(CUSTOMER VARCHAR(80), REGION CHAR(5),
   PURCHASEDATE DATE)
DISTRIBUTE BY HASH (REGION)
```

Views

CREATE VIEW [*schema_name* **.**] *view_name* [**(***column* [**,**...*n*] **)**]

AS *select_statement* [;]

[WITH CHECK OPTION]

- Conditions for updateability of views are similar to SQL Server and Oracle

Materialized Views

- Called materialized query tables and defined with CREATE TABLE

CREATE TABLE table [**(** column_definition [table_constraint] [**,**...*n*] **)]** AS query

Indexes

- Only B+trees

```
CREATE [ UNIQUE  ] INDEX index
  ON table
(column [ ASC | DESC ]
  [, column [ ASC | DESC ] ]...)
```