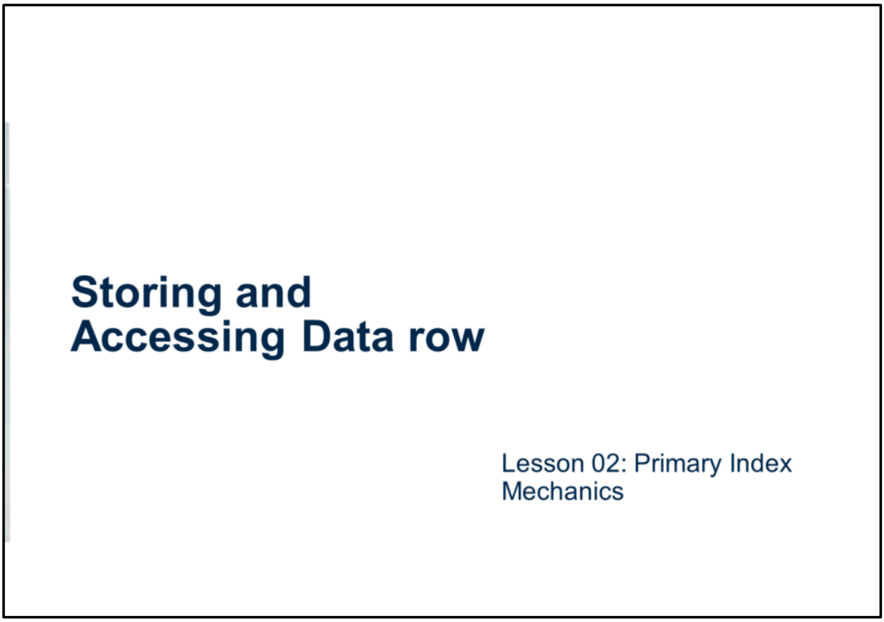
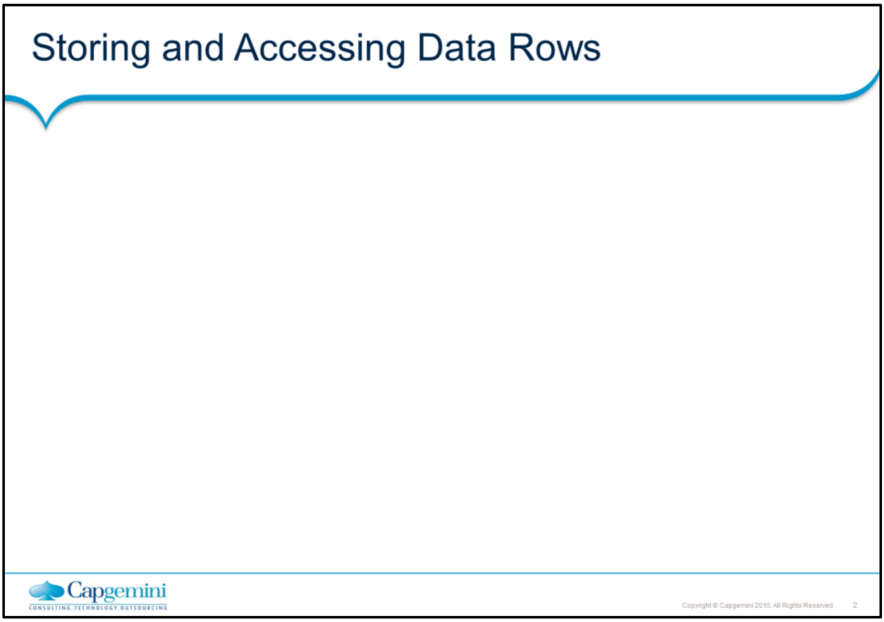
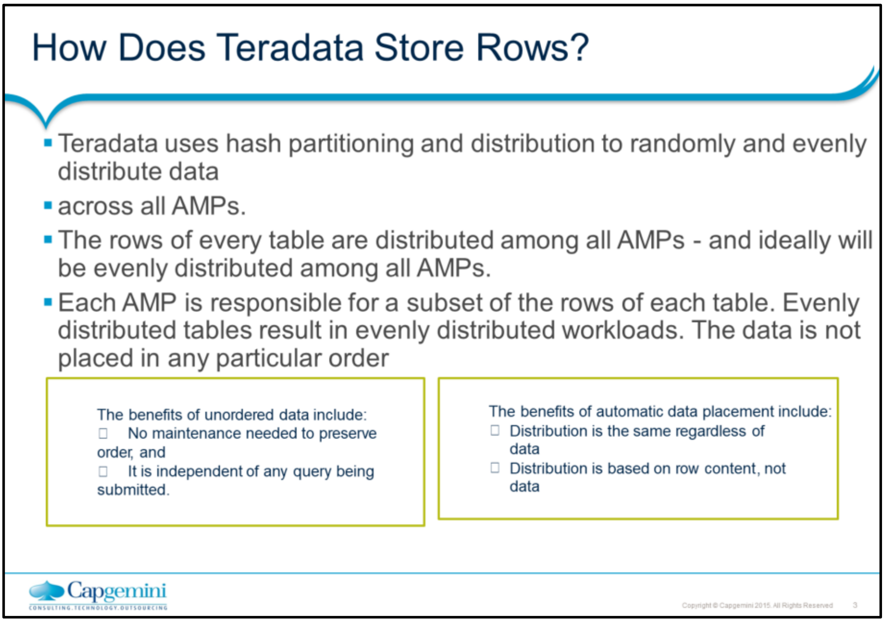


Storing and Accessing Data row

Lesson 02: Primary Index Mechanics



Storing and Accessing Data Rows



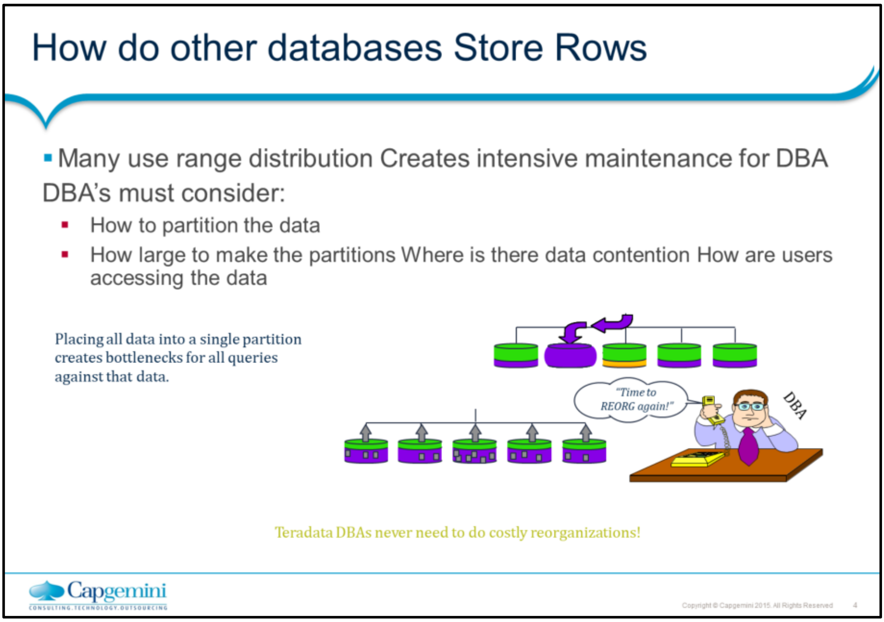
How Does Teradata Store Rows?

- Teradata uses hash partitioning and distribution to randomly and evenly distribute data
- across all AMPs.
- The rows of every table are distributed among all AMPs - and ideally will be evenly distributed among all AMPs.
- Each AMP is responsible for a subset of the rows of each table. Evenly distributed tables result in evenly distributed workloads. The data is not placed in any particular order

The benefits of unordered data include:
□ No maintenance needed to preserve order, and
□ It is independent of any query being submitted.

The benefits of automatic data placement include:
□ Distribution is the same regardless of data
□ Distribution is based on row content, not data

3



How do other databases Store Rows

- Many use range distribution Creates intensive maintenance for DBA

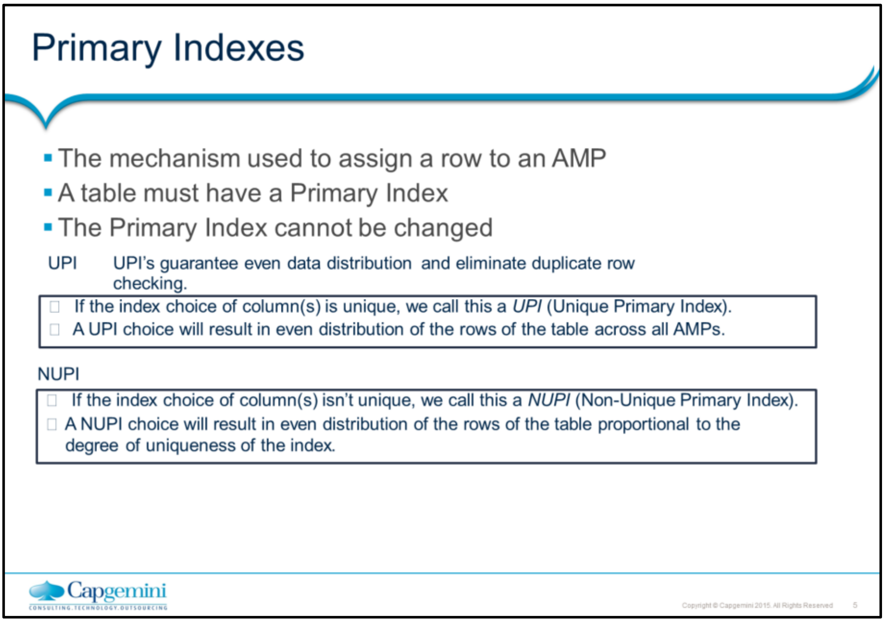
DBA's must consider:

- How to partition the data
- How large to make the partitions Where is there data contention How are users accessing the data

Placing all data into a single partition creates bottlenecks for all queries against that data.

"Time to REORG again!"
DBA

Teradata DBAs never need to do costly reorganizations!



Primary Indexes

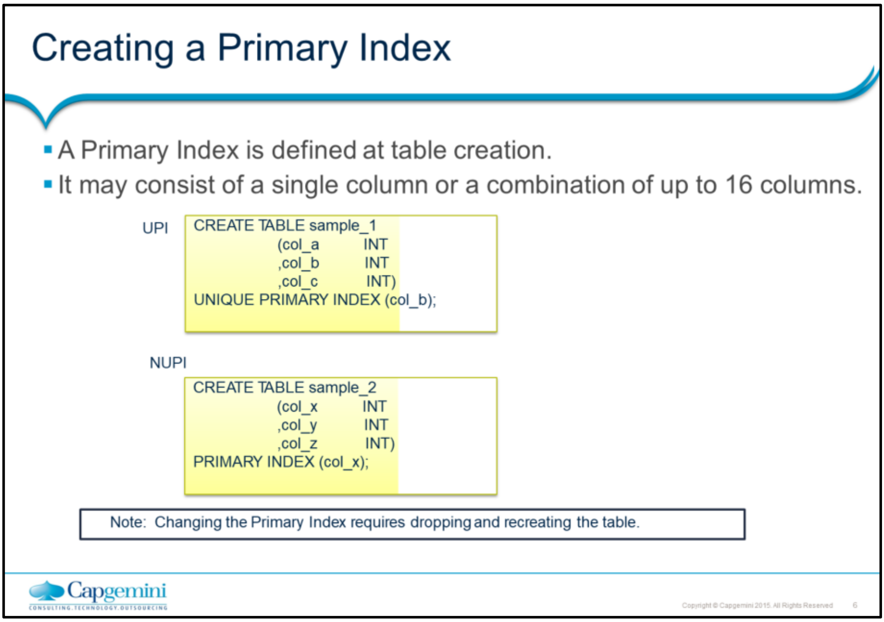
- The mechanism used to assign a row to an AMP
- A table must have a Primary Index
- The Primary Index cannot be changed

UPI UPI's guarantee even data distribution and eliminate duplicate row checking.

□ If the index choice of column(s) is unique, we call this a *UPI* (Unique Primary Index).
□ A UPI choice will result in even distribution of the rows of the table across all AMPs.

NUPI

□ If the index choice of column(s) isn't unique, we call this a *NUPI* (Non-Unique Primary Index).
□ A NUPI choice will result in even distribution of the rows of the table proportional to the degree of uniqueness of the index.



Creating a Primary Index

- A Primary Index is defined at table creation.
- It may consist of a single column or a combination of up to 16 columns.

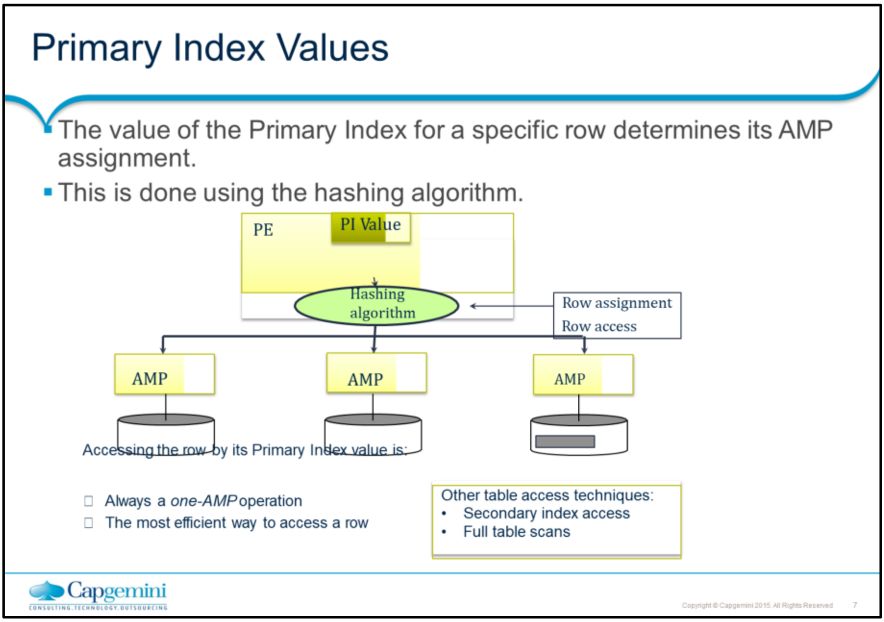
UPI

```
CREATE TABLE sample_1
          (col_a     INT
          ,col_b     INT
          ,col_c     INT)
UNIQUE PRIMARY INDEX (col_b);
```

NUPI

```
CREATE TABLE sample_2
          (col_x     INT
          ,col_y     INT
          ,col_z     INT)
PRIMARY INDEX (col_x);
```

Note: Changing the Primary Index requires dropping and recreating the table.



Primary Index Values

- The value of the Primary Index for a specific row determines its AMP assignment.
- This is done using the hashing algorithm.

PE

PI Value

Hashing algorithm

Row assignment

Row access

AMP

AMP

AMP

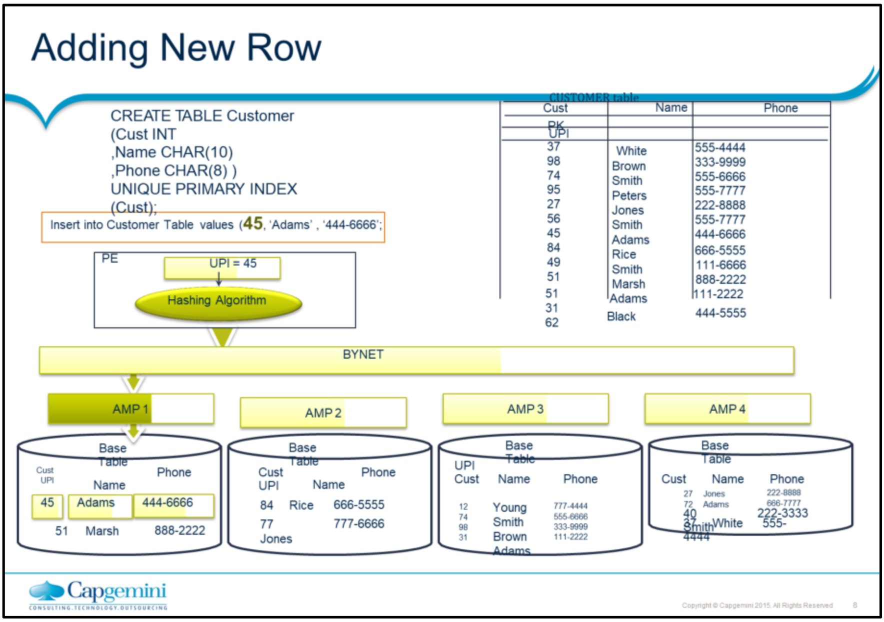
Accessing the row by its Primary Index value is:

□ Always a *one-AMP* operation

□ The most efficient way to access a row

Other table access techniques:

- Secondary index access
- Full table scans



Adding New Row

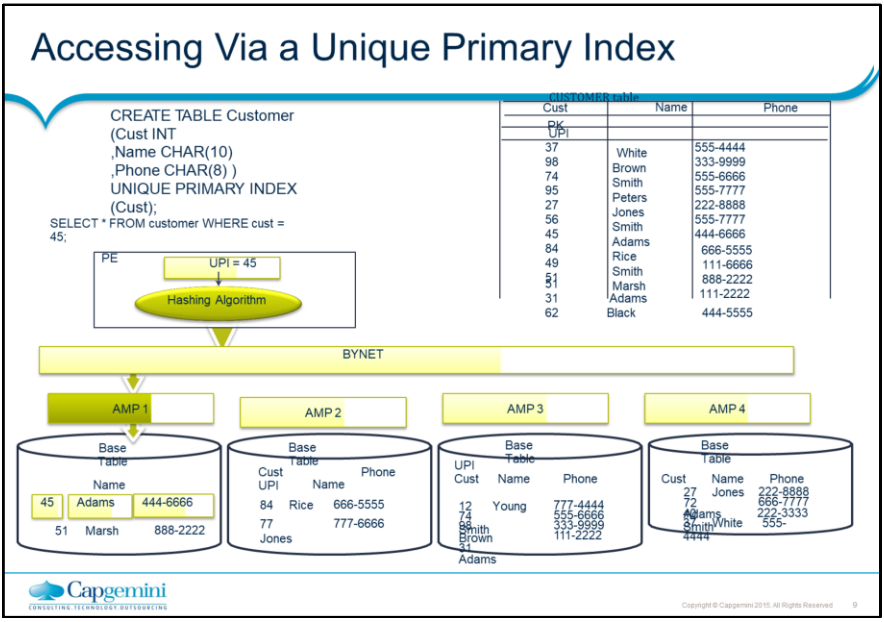
CREATE TABLE Customer
(Cust INT
,Name CHAR(10)
,Phone CHAR(8))
UNIQUE PRIMARY INDEX
(Cust);

Insert into Customer Table values (45, 'Adams' , '444-6666';

CUSTOMER table

Cust	Name	Phone
PK		
UPI		
37	White	555-4444
98	Brown	333-9999
74	Smith	555-6666
95	Peters	555-7777
27	Jones	222-8888
56	Smith	555-7777
45	Adams	444-6666
84	Rice	666-5555
49	Smith	111-6666
51	Marsh	888-2222
51	Adams	111-2222
31	Black	444-5555
62		

8



Accessing Via a Unique Primary Index

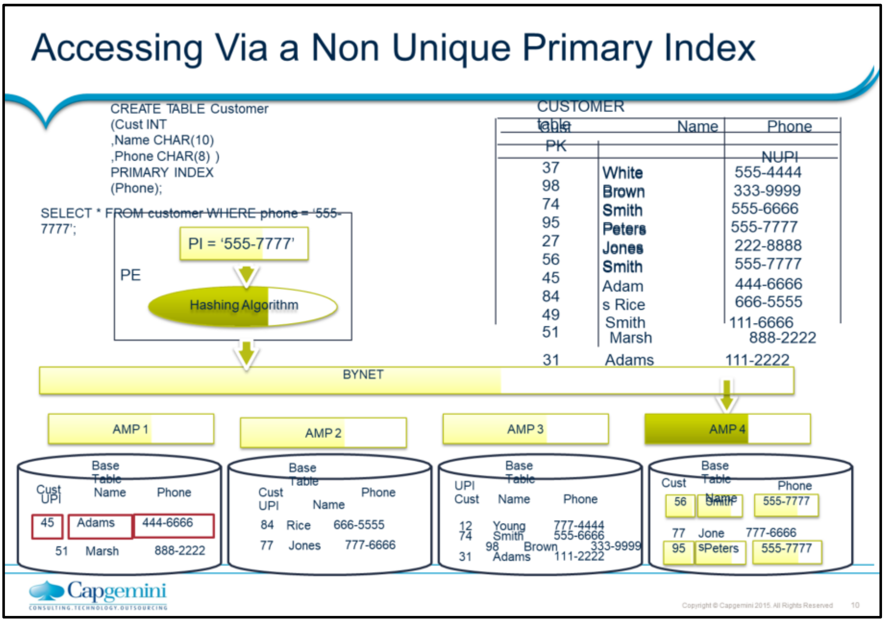
```
CREATE TABLE Customer
(Cust INT
,Name CHAR(10)
,Phone CHAR(8) )
UNIQUE PRIMARY INDEX
(Cust);
SELECT * FROM customer WHERE cust =
45;
```

CUSTOMER table

Cust	Name	Phone
PK		
UPI		
37	White	555-4444
98	Brown	333-9999
74	Smith	555-6666
95	Peters	555-7777
27	Jones	222-8888
56	Smith	555-7777
45	Adams	444-6666
84	Rice	666-5555
49	Smith	111-6666
51 31	Marsh	888-2222
31	Adams	111-2222
62	Black	444-5555

Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

9



Accessing Via a Non Unique Primary Index

```
CREATE  TABLE Customer
(Cust INT
,Name CHAR(10)
,Phone CHAR(8) )
PRIMARY INDEX
(Phone);
```

```
SELECT * FROM customer WHERE phone = '555-7777';
```

CUSTOMER table

Cust PK	Name	Phone NUPI
37	White	555-4444
98	Brown	333-9999
74	Smith	555-6666
95	Peters	555-7777
27	Jones	222-8888
56	Smith	555-7777
45	Adam	444-6666
84	s Rice	666-5555
49	Smith	111-6666
51	Marsh	888-2222
31	Adams	111-2222

AMP 1

Cust UPI	Name	Phone
45	Adams	444-6666
51	Marsh	888-2222

AMP 2

Cust UPI	Name	Phone
84	Rice	666-5555
77	Jones	777-6666

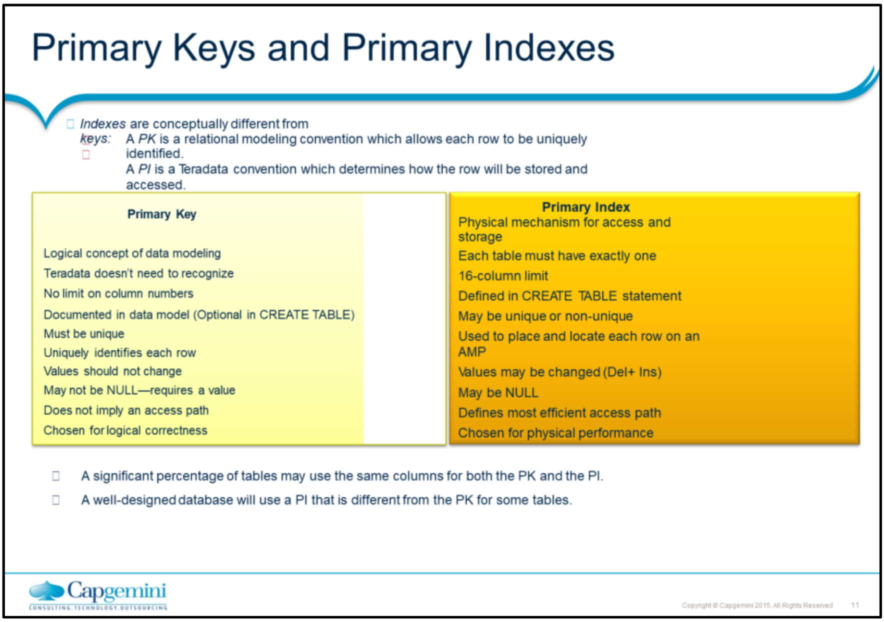
AMP 3

UPI Cust	Name	Phone
12	Young	777-4444
74	Smith	555-6666
98	Brown	333-9999
31	Adams	111-2222

AMP 4

Cust	Name	Phone
56	Smith	555-7777
77	Jone	777-6666
95	sPeters	555-7777

10

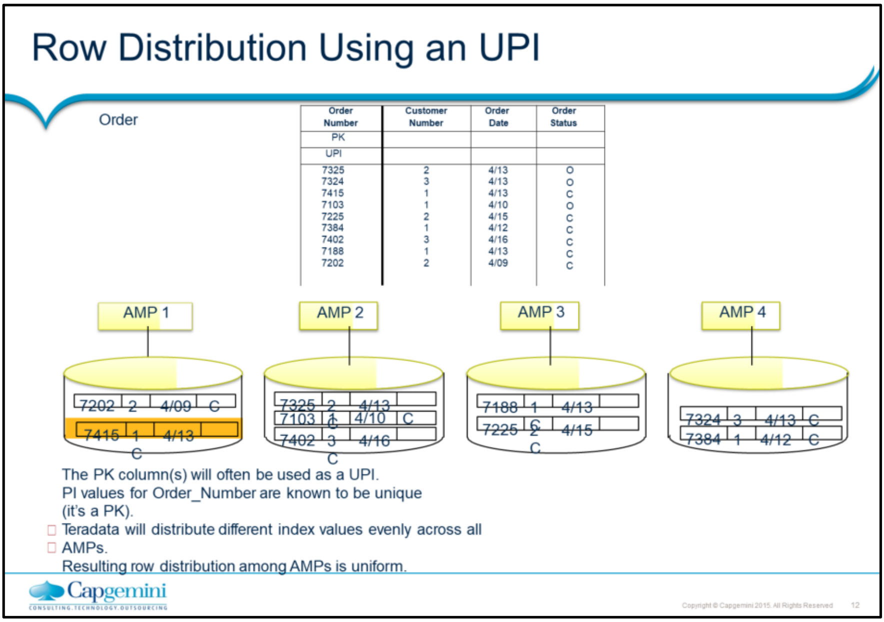


Primary Keys and Primary Indexes

- *Indexes* are conceptually different from *keys:*
  - A *PK* is a relational modeling convention which allows each row to be uniquely identified.
  - A *PI* is a Teradata convention which determines how the row will be stored and accessed.

Primary Key	Primary Index
Logical concept of data modeling	Physical mechanism for access and storage
Teradata doesn't need to recognize	Each table must have exactly one
No limit on column numbers	16-column limit
Documented in data model (Optional in CREATE TABLE)	Defined in CREATE TABLE statement
Must be unique	May be unique or non-unique
Uniquely identifies each row	Used to place and locate each row on an AMP
Values should not change	Values may be changed (Del+ Ins)
May not be NULL—requires a value	May be NULL
Does not imply an access path	Defines most efficient access path
Chosen for logical correctness	Chosen for physical performance

- A significant percentage of tables may use the same columns for both the PK and the PI.
- A well-designed database will use a PI that is different from the PK for some tables.



Row Distribution Using an UPI

Order

Order Number	Customer Number	Order Date	Order Status
PK			
UPI			
7325	2	4/13	O
7324	3	4/13	O
7415	1	4/13	C
7103	1	4/10	O
7225	2	4/15	C
7384	1	4/12	C
7402	3	4/16	C
7188	1	4/13	C
7202	2	4/09	C

AMP 1

7202 2 4/09 C
7415 1 4/13 C

AMP 2

7325 2 4/13
7103 1 C 4/10 C
7402 3 4/16 C

AMP 3

7188 1 4/13
7225 C 2 C 4/15

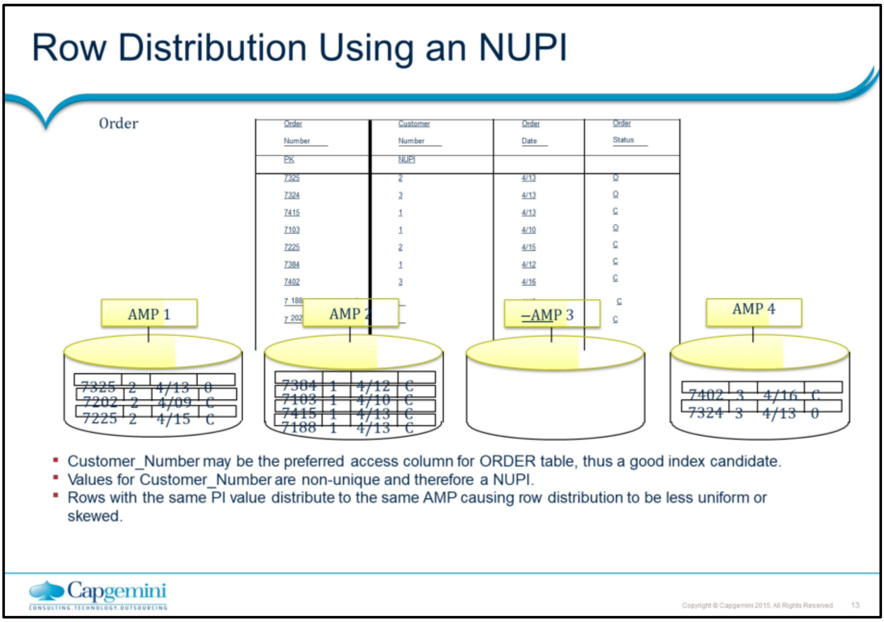
AMP 4

7324 3 4/13 C
7384 1 4/12 C

The PK column(s) will often be used as a UPI.
PI values for Order_Number are known to be unique (it's a PK).

□ Teradata will distribute different index values evenly across all
□ AMPs.

Resulting row distribution among AMPs is uniform.



Row Distribution Using an NUPI

Order

Order Number	Customer Number	Order Date	Order Status
PK	NUPI		
7325	2	4/13	O
7324	3	4/13	O
7415	1	4/13	C
7103	1	4/10	O
7225	2	4/15	C
7384	1	4/12	C
7402	3	4/16	C
7 188	[illegible]	[illegible]	C
7 202	[illegible]	[illegible]	C

AMP 1

7325	2	4/13	O
7202	2	4/09	C
7225	2	4/15	C

AMP 2

7384	1	4/12	C
7103	1	4/10	C
7415	1	4/13	C
7188	1	4/13	C

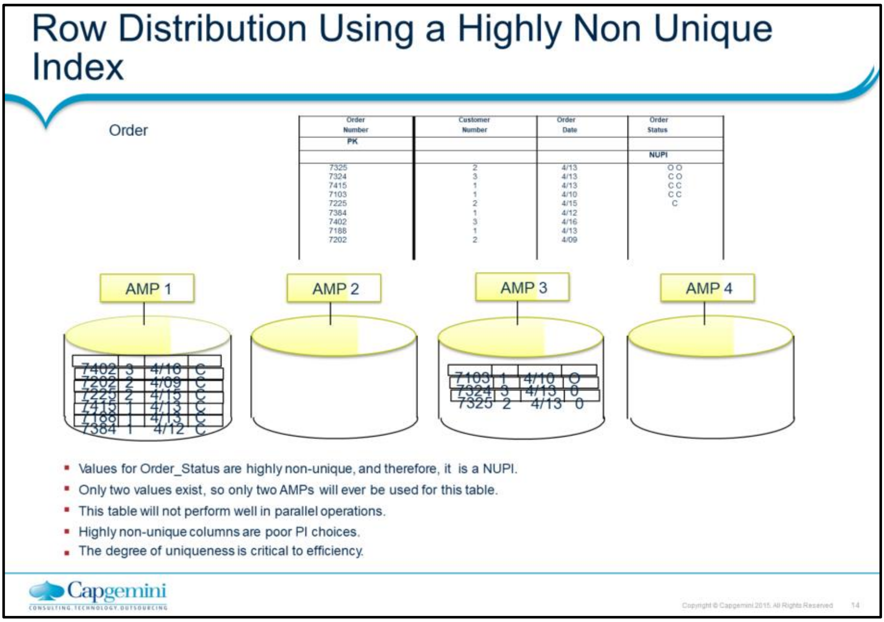
–AMP 3

AMP 4

7402	3	4/16	C
7324	3	4/13	O

- Customer_Number may be the preferred access column for ORDER table, thus a good index candidate.
- Values for Customer_Number are non-unique and therefore a NUPI.
- Rows with the same PI value distribute to the same AMP causing row distribution to be less uniform or skewed.

13



Row Distribution Using a Highly Non Unique Index

Order

Order Number	Customer Number	Order Date	Order Status
PK			
			NUPI
7325	2	4/13	O O
7324	3	4/13	C O
7415	1	4/13	C C
7103	1	4/10	C C
7225	2	4/15	C
7384	1	4/12	
7402	3	4/16	
7188	1	4/13	
7202	2	4/09	

AMP 1

7402	3	4/16	C
7202	2	4/09	C
7225	2	4/15	C
7415	1	4/13	C
7188	1	4/13	C
7384	1	4/12	C

AMP 2

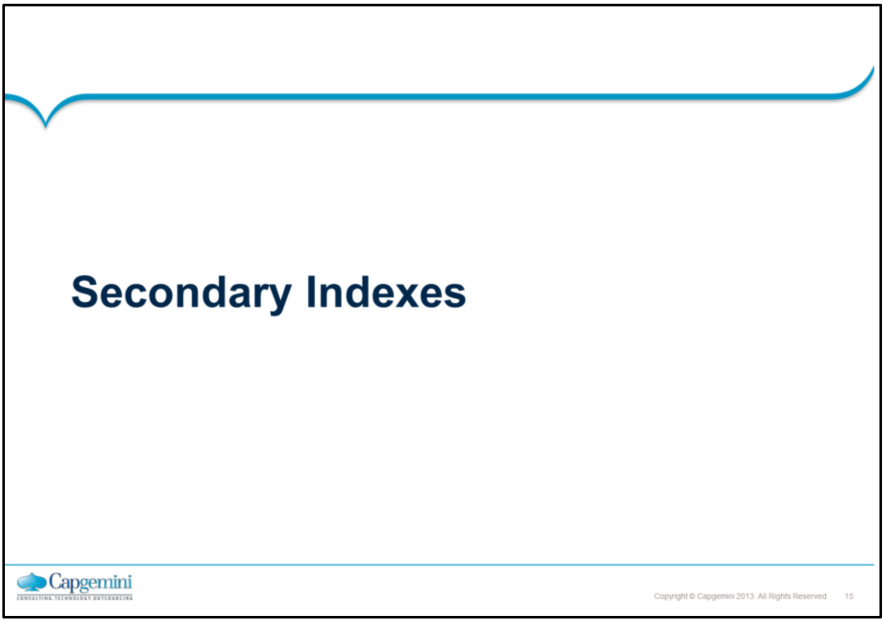
AMP 3

7103	1	4/10	O
7324	3	4/13	0
7325	2	4/13	0

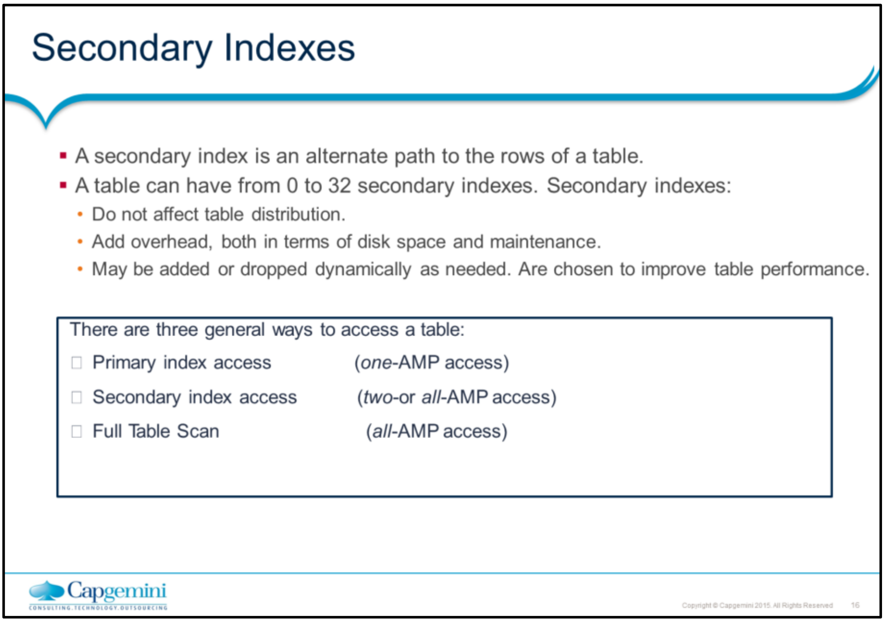
AMP 4

- Values for Order_Status are highly non-unique, and therefore, it is a NUPI.
- Only two values exist, so only two AMPs will ever be used for this table.
- This table will not perform well in parallel operations.
- Highly non-unique columns are poor PI choices.
- The degree of uniqueness is critical to efficiency.

14



Secondary Indexes



Secondary Indexes

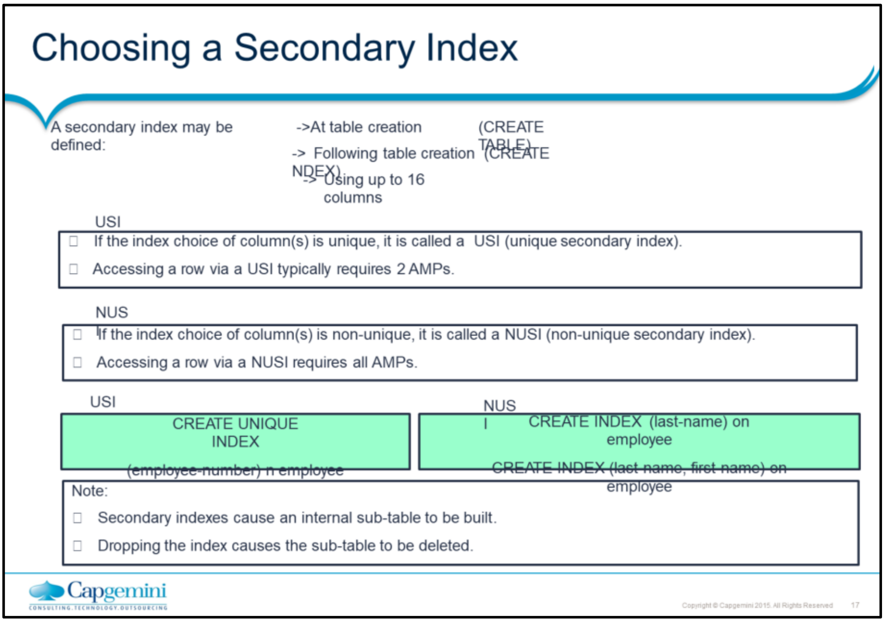
- A secondary index is an alternate path to the rows of a table.
- A table can have from 0 to 32 secondary indexes. Secondary indexes:
  - Do not affect table distribution.
  - Add overhead, both in terms of disk space and maintenance.
  - May be added or dropped dynamically as needed. Are chosen to improve table performance.

There are three general ways to access a table:

□ Primary index access (*one*-AMP access)

□ Secondary index access (*two*-or *all*-AMP access)

□ Full Table Scan (*all*-AMP access)



Choosing a Secondary Index

A secondary index may be defined:

->At table creation (CREATE TABLE)

-> Following table creation (CREATE NDEX)

-> Using up to 16 columns

USI

□ If the index choice of column(s) is unique, it is called a USI (unique secondary index).

□ Accessing a row via a USI typically requires 2 AMPs.

NUS

□ If the index choice of column(s) is non-unique, it is called a NUSI (non-unique secondary index).

□ Accessing a row via a NUSI requires all AMPs.

USI

CREATE UNIQUE INDEX

(employee-number) n employee

NUS
I

CREATE INDEX (last-name) on employee

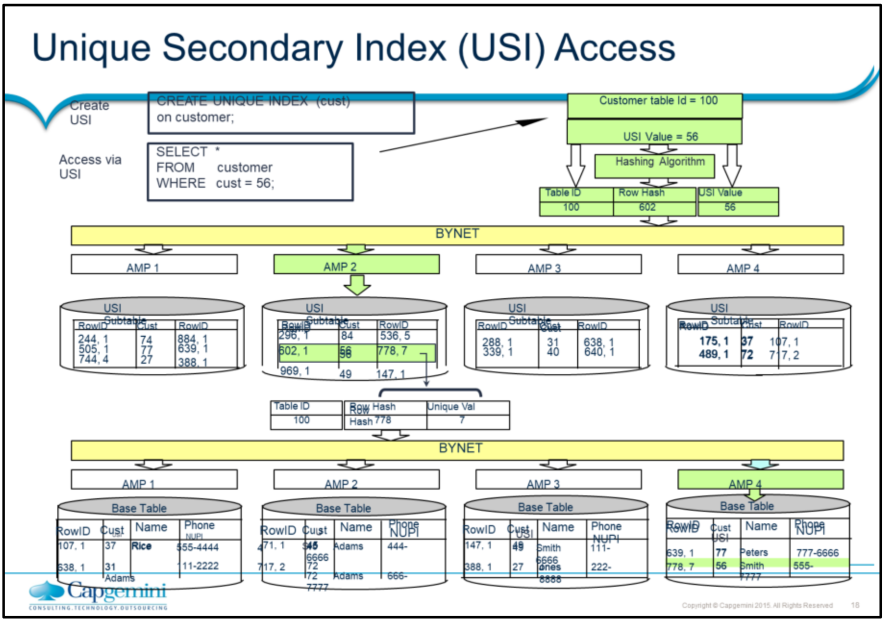
CREATE INDEX (last name, first name) on employee

Note:

□ Secondary indexes cause an internal sub-table to be built.

□ Dropping the index causes the sub-table to be deleted.

17



Unique Secondary Index (USI) Access

Create USI

```
CREATE UNIQUE INDEX (cust)
on customer;
```

Access via USI

```
SELECT *
FROM     customer
WHERE   cust = 56;
```

Table ID	Row Hash	USI Value
100	602	56

USI Subtable (AMP 1)

RowID	Cust	RowID
244, 1	74	884, 1
505, 1	77	639, 1
744, 4	27	388, 1

USI Subtable (AMP 2)

RowID	Cust	RowID
296, 1	84	536, 5
602, 1	56	778, 7
969, 1	49	147, 1

USI Subtable (AMP 3)

RowID	Cust	RowID
288, 1	31	638, 1
339, 1	40	640, 1

USI Subtable (AMP 4)

RowID	Cust	RowID
175, 1	37	107, 1
489, 1	72	717, 2

Table ID	Row Hash	Unique Val
100	778	7

Base Table (AMP 1)

RowID	Cust USI	Name	Phone NUPI
107, 1	37	Rice	555-4444
638, 1	31	Adams	111-2222

Base Table (AMP 2)

RowID	Cust USI	Name	Phone NUPI
471, 1	45	Adams	444-6666
717, 2	72	Adams	666-7777

Base Table (AMP 3)

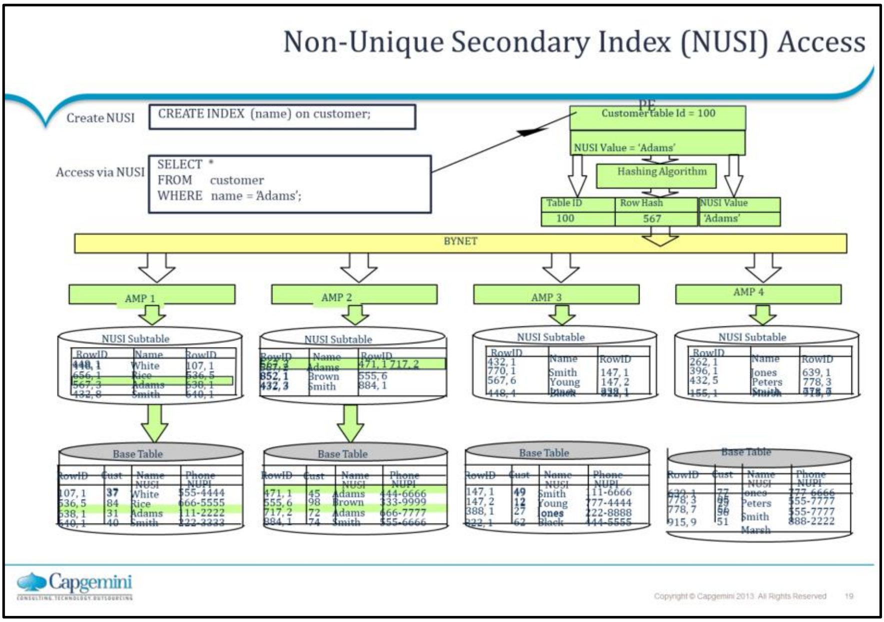
RowID	Cust USI	Name	Phone NUPI
147, 1	49	Smith	111-6666
388, 1	27	Jones	222-8888

Base Table (AMP 4)

RowID	Cust USI	Name	Phone NUPI
639, 1	77	Peters	777-6666
778, 7	56	Smith	555-7777

Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

18



Non-Unique Secondary Index (NUSI) Access

Create NUSI

```
CREATE INDEX (name) on customer;
```

Access via NUSI

```
SELECT *
FROM     customer
WHERE  name = 'Adams';
```

Table ID	Row Hash	NUSI Value
100	567	'Adams'

AMP 1 – NUSI Subtable

RowID	Name	RowID
448, 1	White	107, 1
656, 1	Rice	536, 5
567, 3	Adams	638, 1
432, 8	Smith	640, 1

AMP 2 – NUSI Subtable

RowID	Name	RowID
567, 2	Adams	471, 1 717, 2
852, 1	Brown	555, 6
432, 3	Smith	884, 1

AMP 3 – NUSI Subtable

RowID	Name	RowID
432, 1	Smith	147, 1
770, 1	Young	147, 2
567, 6	Jones	388, 1
448, 4	Black	822, 1

AMP 4 – NUSI Subtable

RowID	Name	RowID
262, 1	Jones	639, 1
396, 1	Peters	778, 3
432, 5	Smith	778, 7
155, 1	Marsh	915, 9

AMP 1 – Base Table

RowID	Cust	Name NUSI	Phone NUPI
107, 1	37	White	555-4444
536, 5	84	Rice	666-5555
638, 1	31	Adams	111-2222
640, 1	40	Smith	222-3333

AMP 2 – Base Table

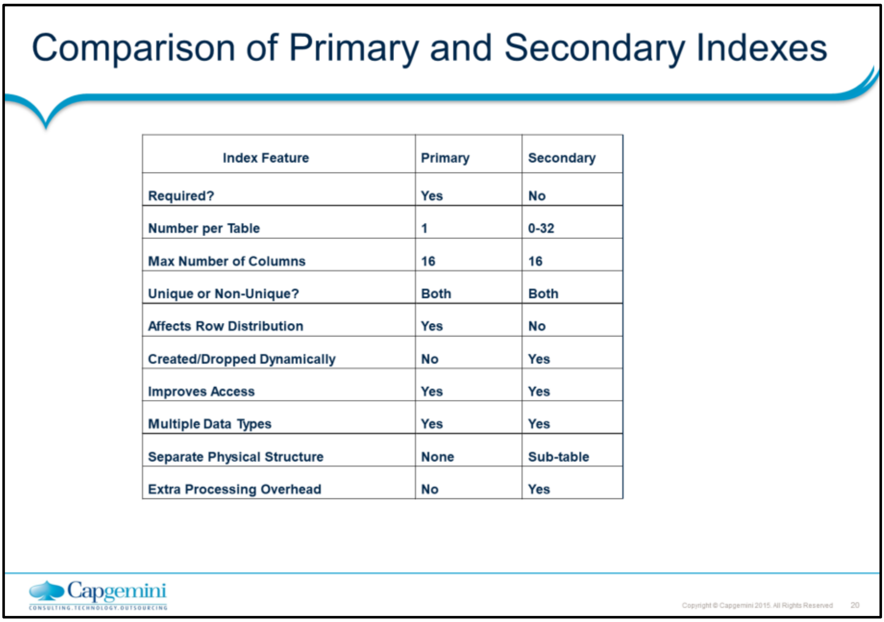
RowID	Cust	Name NUSI	Phone NUPI
471, 1	45	Adams	444-6666
555, 6	98	Brown	333-9999
717, 2	72	Adams	666-7777
884, 1	74	Smith	555-6666

AMP 3 – Base Table

RowID	Cust	Name NUSI	Phone NUPI
147, 1	49	Smith	111-6666
147, 2	12	Young	777-4444
388, 1	27	Jones	222-8888
822, 1	62	Black	444-5555

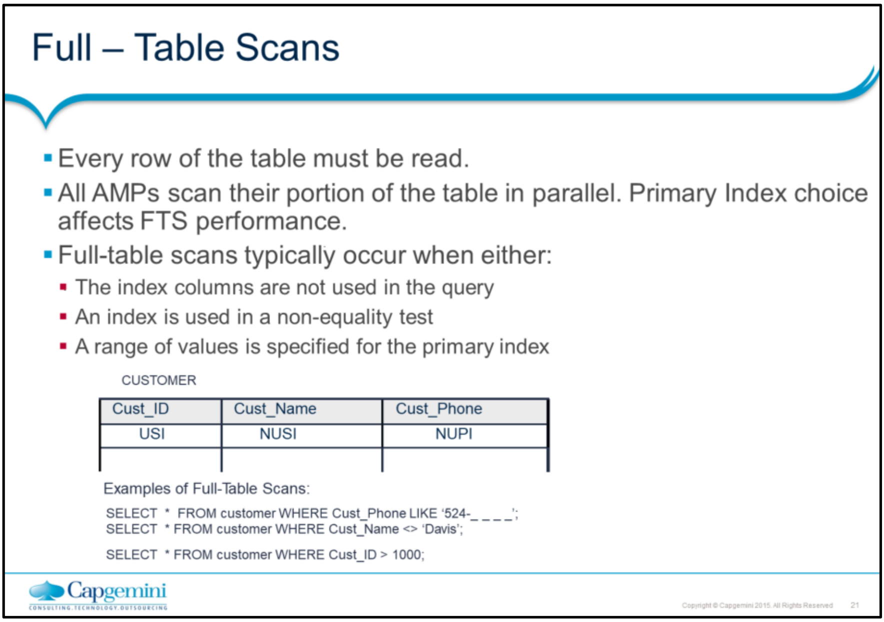
AMP 4 – Base Table

RowID	Cust	Name NUSI	Phone NUPI
639, 1	77	Jones	777-6666
778, 3	95	Peters	555-7777
778, 7	56	Smith	555-7777
915, 9	51	Marsh	888-2222



Comparison of Primary and Secondary Indexes

Index Feature	Primary	Secondary
Required?	Yes	No
Number per Table	1	0-32
Max Number of Columns	16	16
Unique or Non-Unique?	Both	Both
Affects Row Distribution	Yes	No
Created/Dropped Dynamically	No	Yes
Improves Access	Yes	Yes
Multiple Data Types	Yes	Yes
Separate Physical Structure	None	Sub-table
Extra Processing Overhead	No	Yes



Full – Table Scans

- Every row of the table must be read.
- All AMPs scan their portion of the table in parallel. Primary Index choice affects FTS performance.
- Full-table scans typically occur when either:
  - The index columns are not used in the query
  - An index is used in a non-equality test
  - A range of values is specified for the primary index

CUSTOMER

Cust_ID	Cust_Name	Cust_Phone
USI	NUSI	NUPI

Examples of Full-Table Scans:

```
SELECT  *  FROM customer WHERE Cust_Phone LIKE '524-_ _ _ _';
SELECT  * FROM customer WHERE Cust_Name <> 'Davis';

SELECT  * FROM customer WHERE Cust_ID > 1000;
```