*** Create a range partitioned index (example taken straight out of the manuals)

*** Note that the time_id is the partitioned column

```
SQL> CREATE TABLE range_sales
  2       ( prod_id        NUMBER(6)
  3       , cust_id        NUMBER
  4       , time_id        DATE
  5       , channel_id     CHAR(1)
  6       , promo_id       NUMBER(6)
  7       , quantity_sold  NUMBER(3)
  8       , amount_sold    NUMBER(10,2)
  9       )
 10  PARTITION BY RANGE (time_id)
 11    (PARTITION SALES_Q1_1998 VALUES LESS THAN (TO_DATE('01-APR-
1998','DD-MON-YYYY')),
 12     PARTITION SALES_Q2_1998 VALUES LESS THAN (TO_DATE('01-JUL-
1998','DD-MON-YYYY')),
 13     PARTITION SALES_Q3_1998 VALUES LESS THAN (TO_DATE('01-OCT-
1998','DD-MON-YYYY')),
 14     PARTITION SALES_Q4_1998 VALUES LESS THAN (TO_DATE('01-JAN-
1999','DD-MON-YYYY')),
 15     PARTITION SALES_Q1_1999 VALUES LESS THAN (TO_DATE('01-APR-
1999','DD-MON-YYYY')),
 16     PARTITION SALES_Q2_1999 VALUES LESS THAN (TO_DATE('01-JUL-
1999','DD-MON-YYYY')),
 17     PARTITION SALES_Q3_1999 VALUES LESS THAN (TO_DATE('01-OCT-
1999','DD-MON-YYYY')),
 18     PARTITION SALES_Q4_1999 VALUES LESS THAN (TO_DATE('01-JAN-
2000','DD-MON-YYYY')),
 19     PARTITION SALES_Q1_2000 VALUES LESS THAN (TO_DATE('01-APR-
2000','DD-MON-YYYY')),
 20     PARTITION SALES_Q2_2000 VALUES LESS THAN (TO_DATE('01-JUL-
2000','DD-MON-YYYY')),
 21     PARTITION SALES_Q3_2000 VALUES LESS THAN (TO_DATE('01-OCT-
2000','DD-MON-YYYY')),
 22     PARTITION SALES_Q4_2000 VALUES LESS THAN (MAXVALUE))
 23  ;

Table created.
```

*** Attempt to create a LOCAL Unique index on just the prod_id column

*** Note it fails as the partitioned column (time_id) is not part of the Unique index

```
SQL> CREATE UNIQUE INDEX range_sales_i ON range_sales(prod_id) LOCAL;
CREATE UNIQUE INDEX range_sales_i ON range_sales(prod_id) LOCAL
                                    *
ERROR at line 1:
ORA-14039: partitioning columns must form a subset of key columns of a
UNIQUE index
```

*** Note that you can create a LOCAL Non-unique index on just the prod_id column ...

```
SQL> CREATE INDEX range_sales_i ON range_sales(prod_id) LOCAL;

Index created.
```

*** BUT BUT BUT, you still can't use this Non-Unique index to police a PK (or Unique) constraint as it doesn't include the partitioned column(s) as part of the constraint

```
SQL> ALTER TABLE range_sales ADD PRIMARY KEY(prod_id);
ALTER TABLE range_sales ADD PRIMARY KEY(prod_id)
*
ERROR at line 1:
ORA-01408: such column list already indexed
```

*** Oracle wants to create a GLOBAL index by default but can't as the prod_id column is already indexed, hence the error

*** So not only can you not create a LOCAL Unique index that doesn't include the partitioned columns, but you can't create a LOCAL Non-Unique index to police such a constraint either

```
SQL> DROP INDEX range_sales_i;

Index dropped.
```

*** Create a GLOBAL Non-Unique or Unique index and all is fine ...

*** First, Non-Unique Global Index

```
SQL> CREATE INDEX range_sales_i ON range_sales(prod_id) GLOBAL;

Index created.

SQL> ALTER TABLE range_sales ADD PRIMARY KEY(prod_id);

Table altered.
```

```
SQL> ALTER TABLE range_sales DROP PRIMARY KEY;

Table altered.

SQL> DROP INDEX range_sales_i;

Index dropped.
```

*** Next, Unique Global Index

```
SQL> CREATE UNIQUE INDEX range_sales_i ON range_sales(prod_id) GLOBAL;

Index created.
```

```
SQL> ALTER TABLE range_sales ADD PRIMARY KEY(prod_id);

Table altered.


*** Or add the partitioned column to the constraint and you can create
either a Unique or Non-Unique Local index

SQL> ALTER TABLE range_sales DROP PRIMARY KEY;

Table altered.

SQL> DROP INDEX range_sales_i;

Index dropped.


*** First, Unique Local Index, which must include the partitioned column
(time_id) in both index and constraint


SQL> CREATE UNIQUE INDEX range_sales_i ON range_sales(prod_id, time_id)
LOCAL;

Index created.

SQL> ALTER TABLE range_sales ADD PRIMARY KEY(prod_id, time_id);

Table altered.


SQL> ALTER TABLE range_sales DROP PRIMARY KEY;

Table altered.

SQL> DROP INDEX range_sales_i;

Index dropped.


*** Last, Non-Unique Local Index, which must also include the partitioned
column (time_id) in both index and constraint


SQL> CREATE INDEX range_sales_i ON range_sales(prod_id, time_id) LOCAL;

Index created.

SQL> ALTER TABLE range_sales ADD PRIMARY KEY(prod_id, time_id);

Table altered.
```