Oracle PRIMARY KEY

A primary key is a column of a combination of columns in a table that uniquely identifies a row in the table.

def:a **primary key** is a single field or combination of fields that uniquely defines a record. None of the fields that are part of the primary key can contain a null value. A table can have only one
primary key.

The following are rules that make a column a primary key:

- A primary key column cannot contain a NULL value or an empty string.
- A primary key value must be unique within the entire table.
- A primary key value should not be changed over time.
- In Oracle, a primary key can not contain more than 32 columns.
- A primary key can be defined in either a CREATE TABLE statement or an ALTER TABLE statement.

According to these rules, the following are the recommendations for the primary keys:

- First, the primary key should be meaningless. Sometimes, you may want use meaningful data, which considers being unique, for the primary keys e.g., social security number (SSN), vehicle identification number (VIN), email, and phone number. However, you don’t know when the email or phone number changes or is reused by another person. In such cases, it will create many data problems. In the database world, the artificialkeys are known as *surrogate keys* which are as opposed to *natural* primary keys.
- Second, the primary keys should be compact. The primary keys typically are numeric because Oracle typically processes numbers faster than any other data types.

To create a primary key in a table, you use the `PRIMARY KEY` constraint.

Creating a primary key that consists of one column

```
CREATE TABLE purchase_orders (
    po_nr NUMBER PRIMARY KEY,
    vendor_id NUMBER NOT NULL,
    po_status NUMBER(1,0) NOT NULL,
    created_at TIMESTAMP WITH TIME ZONE NOT NULL
);
```

```
CREATE TABLE purchase_orders (
    po_nr NUMBER,
    vendor_id NUMBER NOT NULL,
    po_status NUMBER(1,0) NOT NULL,
    created_at TIMESTAMP WITH TIME ZONE NOT NULL,
    CONSTRAINT pk_purchase_orders PRIMARY KEY(po_nr)
);
```

we explicitly assigned the `PRIMARY KEY` constraint a name `pk_purchase_orders`

Creating a primary key that consists of multiple columns

```
CREATE TABLE purchase_order_items (
    po_nr NUMBER NOT NULL,
    item_nr NUMBER NOT NULL,
    product_id NUMBER NOT NULL,
    quantity NUMBER NOT NULL,
    purchase_unit NUMBER NOT NULL,
    buy_price NUMBER (9,2) NOT NULL,
    delivery_date DATE,
    PRIMARY KEY (po_nr, item_nr)
);
```

did not use the `CONSTRAINT` clause to explicitly assign the `PRIMARY KEY` constraint a name. Therefore, Oracle implicitly assigned the primary key constraint a system-generated name such as `SYS_C0010617`.

Adding a primary key to a table

```
ALTER TABLE table_name
ADD CONSTRAINT constraint_name
PRIMARY KEY (column1, column2, ...);
```

```
CREATE TABLE vendors (
    vendor_id NUMBER,
    vendor_name VARCHAR2(255) NOT NULL,
    address VARCHAR2(255) NOT NULL
);

ALTER TABLE vendors
ADD CONSTRAINT pk_vendors PRIMARY KEY (vendor_id);
```

Dropping an Oracle `PRIMARY KEY` constraint

```
ALTER TABLE vendors
DROP CONSTRAINT pk_vendors;
```

```
ALTER TABLE vendors
DROP PRIMARY KEY;
```

Enable / Disable an Oracle `PRIMARY KEY` constraint

To improve the performance when loading a large amount of data into a table or updating mass data, you can temporarily disable the `PRIMARY KEY` constraint.

```
ALTER TABLE purchase_orders
DISABLE CONSTRAINT pk_purchase_orders;

#OR

ALTER TABLE purchase_orders
DISABLE PRIMARY KEY;
```

```
ALTER TABLE purchase_orders
ENABLE CONSTRAINT pk_purchase_orders;

#OR

ALTER TABLE purchase_orders
ENABLE PRIMARY KEY;
```