PRIMARY KEY

The primary key is a simple column

```
create table person (
person_id bigint PRIMARY KEY,
city text);
```

RowKey: person_id

PRIMARY KEY (COMPOSITE/COMPOUND)

The composite key consists of multiple columns

```
create table person (person_id bigint,transaction_id bigint, city text,
datetime timestamp, PRIMARY KEY(person_id, transaction_id));
```

RowKey: person_id

person_id is a **partition key**
transaction_id **is a clustering key**

COMPOSITE PRIMARY KEY

```
create table person (person_id bigint,transaction_id bigint, city text,
datetime timestamp, PRIMARY KEY((person_id, transaction_id),city));
```

RowKey: person_id + transaction_id

- The **Partition Key** is responsible for data distribution across your nodes.
- The **Clustering Key** is responsible for data sorting within the partition.
- The **Primary Key** is equivalent to the **Partition Key** in a single-field-key table (i.e. **Simple**).
- The **Composite/Compound Key** is just any multiple-column key

EXAMPLE

Step 1 - Create table with Composite Primary Key

```
create table location (
vehicle_id text,
date text,
time timestamp,
lat double,
long double,
PRIMARY KEY ((vehicle_id, date),time))
WITH CLUSTERING ORDER BY (time DESC);
```

Step 2 - Load data

Keep following data as a csv

```
06TOY12,2014-01-01,2014-01-01 08:50:00,44.749,-67,251
06BA2013,2015-01-01,2015-01-01 08:50:00,45.749,-66,251
34BA2013,2016-01-01,2017-01-01 08:50:00,42.749,-62,251
```

Code

```
COPY location (vehicle_id,date, time, lat,long) FROM '/home/ubuntu/lat.csv'
WITH HEADER=false AND delimiter='|';
```

```
(0 rows)
cqlsh:user_transaction> COPY location (vehicle_id,date, time, lat,long) FROM '/h
ome/ubuntu/lat.csv' WITH HEADER=false AND delimiter='|';
Using 1 child processes

Starting copy of user_transaction.location with columns [vehicle_id, date, time,
 lat, long].
Processed: 3 rows; Rate:       5 rows/s; Avg. rate:       7 rows/s
3 rows imported from 1 files in 0.404 seconds (0 skipped).
```