Challenge 1

Below is the DDL that was used to create a table in PostgreSQL 12. The table now contains 15 million rows in production. We want to change this table to be partitioned by month on the ts column. The existing data is all from 2020. We can make the change in after-hours maintenance, but we do not want to take the service completely down, ideally. The service only executes inserts and selects against this table. Inserts always use the current timestamp. Please write a SQL script that would affect the change while being the least impactful to the running service. Provide an explanation of why you chose the approach you did.

create table foo.bar

( id bigint not null generated always as identity primary key

, ts timestamptz not null default now()

, baz text not null

, buz text

, biz text

);

create unique index on foo.bar using btree (baz);

**Solution:**

By seeing the table structure and timestamp column is monotonically increasing timestamp data and having the unique index and constraints we will be going with declarative partitioning. We will load the data into a new table initially and during the maintainence Window (Downtime) we replace the existing table.

**Pre-work:**

* Create a new table foo.bar\_partition with partitions.
* Copy the data to the new table from the existing table until some specific timestamp.
* Create the unique index as available along with the partition key else create a normal binary index. (Verify this with the application effect about the need of unique key).

**During after-hours Maintenance:**

* Rename the old table to foo.bar\_backup\_02222021;
* Apply the incremental changes from specific timestamp choosen and consider remaining all rows.
* Rename the partition table to the base table before foo.bar

**Pre-Work Steps**

1. **Create the partition table with**

*CREATE TABLE foo.bar\_partition*

*(*

*id bigint GENERATED ALWAYS AS IDENTITY (MAXVALUE 9223372036854775807),*

*ts timestamptz DEFAULT now() NOT NULL,*

*baz text NOT NULL,*

*buz text,*

*biz text*

*) PARTITION BY RANGE (ts);*

*CREATE TABLE bar\_2020\_01 PARTITION OF foo.bar\_partition FOR VALUES FROM ('2020-01-01') TO ('2020-01-31');*

*CREATE TABLE bar\_2020\_02 PARTITION OF foo.bar\_partition FOR VALUES FROM ('2020-02-01') TO ('2020-02-28');*

*CREATE TABLE bar\_2020\_03 PARTITION OF foo.bar\_partition FOR VALUES FROM ('2020-03-01') TO ('2020-03-31');*

*CREATE TABLE bar\_2020\_04 PARTITION OF foo.bar\_partition FOR VALUES FROM ('2020-04-01') TO ('2020-04-30');*

*CREATE TABLE bar\_2020\_05 PARTITION OF foo.bar\_partition FOR VALUES FROM ('2020-05-01') TO ('2020-05-31');*

*CREATE TABLE bar\_2020\_06 PARTITION OF foo.bar\_partition FOR VALUES FROM ('2020-06-01') TO ('2020-06-30');*

*CREATE TABLE bar\_2020\_07 PARTITION OF foo.bar\_partition FOR VALUES FROM ('2020-07-01') TO ('2020-07-31');*

*CREATE TABLE bar\_2020\_08 PARTITION OF foo.bar\_partition FOR VALUES FROM ('2020-08-01') TO ('2020-08-31');*

*CREATE TABLE bar\_2020\_09 PARTITION OF foo.bar\_partition FOR VALUES FROM ('2020-09-01') TO ('2020-09-30');*

*CREATE TABLE bar\_2020\_10 PARTITION OF foo.bar\_partition FOR VALUES FROM ('2020-10-01') TO ('2020-10-31');*

*CREATE TABLE bar\_2020\_11 PARTITION OF foo.bar\_partition FOR VALUES FROM ('2020-11-01') TO ('2020-11-30');*

*CREATE TABLE bar\_2020\_12 PARTITION OF foo.bar\_partition FOR VALUES FROM ('2020-12-01') TO ('2020-12-31');*

1. Load the data from the base table to partition table

*insert into foo.bar\_pr\_test(ts, baz,buz,biz)*

*select ts,baz,buz,biz*

*from foo.bar where ts < '2020-12-31 12:00:00.000000+00'*

1. Create the Unique index

*create unique index on foo.bar\_partition using btree (baz,ts);*

**After Hours Maintenance:**

1. Rename the old table to foo.bar\_backup

*Alter table foo.bar rename to bar\_backup\_0222021;*

1. Load the incremental changes on the table

*insert into foo.bar\_partition(ts, baz,buz,biz)*

*select ts,baz,buz,biz*

*from foo.bar where ts >= '2020-12-31 12:00:00.000000+00'*

1. Rename the partitioned table

*Alter table foo.bar\_partition rename to foo.bar*

**Pitfalls/Issues encountered:**

ERROR: unique constraint on partitioned table must include all partitioning columns DETAIL: UNIQUE constraint on table "bar\_partition" lacks column "ts" which is part of the partition key. SQL state: 0A000

**To over come the problem we need to add the partition index to the unique index.**

**The automatic interval partition is available with EDB Postgres and not available with the 2nd Quadrant Postgres 12**

**Notes:**

To add data to the table used the below pl/pgsql block to add the rows to the table.

*do*

*$$*

*begin*

*for i in 1..1000000*

*loop*

*insert into foo.bar (baz) values ('The value of the text is ' || i);*

*commit;*

*end loop;*

*end;*

*$$*