**Assignment 3**

# Index Tuning

**Database Tuning**

New Group 8
Frauenschuh Florian, 12109584
Lindner Peter, 12101607
Weilert Alexander, 12119653

April 30, 2024

**Database system and version:** `Postgres 14.11` with driver `postgresql 42.7.3`

## 1 Index Data Structures

Which index data structures (e.g., $B^+$ tree index) are supported?

[Your answer goes here ...]

## 2 Clustering Indexes

Discuss how the system supports clustering indexes, in particular:

**a)** How do you create a clustering index on `ssnum`? Show the query.[1]

First, we assume the table `Employee` to be the same as in the previous assignment:

```
CREATE TABLE IF NOT EXISTS Employee (
    ssnum INTEGER PRIMARY KEY,
    name VARCHAR(64) UNIQUE NOT NULL,
    manager VARCHAR(64),
    dept VARCHAR(64),
    salary INTEGER,
    numfriends INTEGER)
```

We note that `ssnum` is the primary key of the table, hence Postgres automatically creates an B-Tree based, *unique index* for it [1].

Now to be able to create a clustering index, we need an index to begin with. As discussed earlier, such an index already exists for `ssnum`. Thus, we can *cluster* this index according to [2] by performing:

```
CLUSTER Employee USING idx_ssnum
```

[1] Give the queries for creating a hash index *and* a $B^+$ tree index if both of them are supported.

Here we assumed the index on `ssnum` to be named `idx_ssnum`.

If we had to create such an index by ourselves, one could accomplish this by performing:

```
CREATE UNIQUE INDEX idx_ssnum ON Employee [using btree] (ssnum)
```

The additional command in parentheses `using btree` is optional, since it already is the default for Postgres (and Postgres only supports unique indexes using B-Trees) [1].

**b)** Are clustering indexes on non-key attributes supported, e.g., on `name`? Show the query.

Yes, clustering indexes are also supported on non-key attributes.

As mentioned earlier, we first need an index on name to cluster it:

```
CREATE INDEX idx_name on Employee (name)
```

Followed by that, we can now cluster it by performing:

```
CLUSTER Employee USING idx_name
```

**c)** Is the clustering index dense or sparse?

[Your answer goes here ...]

**d)** How does the system deal with overflows in clustering indexes? How is the fill factor controlled?

[Your answer goes here ...]

**e)** Discuss any further characteristics of the system related to clustering indexes that are relevant to a database tuner.

[Your answer goes here ...]

## 3 Non-Clustering Indexes

Discuss how the system supports non-clustering indexes, in particular:

**a)** How do you create a combined, non-clustering index on `(dept,salary)`? Show the query.[1]

[Your answer goes here ...]

```
[Your SQL query goes here ...]
```

**b)** Can the system take advantage of covering indexes? What if the index covers the query, but the condition is not a prefix of the attribute sequence `(dept,salary)`?

[Your answer goes here ...]

**c)** Discuss any further characteristics of the system related to non-clustering indexes that are relevant to a database tuner.

[Your answer goes here ...]

## 4 Key Compression and Page Size

If your system supports $B^+$ trees, what kind of key compression (if any) is supported? How large is the default disk page? Can it be changed?

[Your answer goes here ...]

## Time Spent on this Assignment

Time in hours per person:

- Florian Frauenschuh:
- Peter Lindner:
- Alexander Weilert:

## References

[1] PostgreSQL. *Unique Indexes*. Accessed: 2023-04-30. 2023. URL: https://www.postgresql.org/docs/14/indexes-unique.html.

[2] PostgreSQL. *CLUSTER*. Accessed: 2023-04-30. 2023. URL: https://www.postgresql.org/docs/14/sql-cluster.html.