Save COPY error details to an existing table

jian he jian.universality@gmail.com

commitfest: https://commitfest.postgresql.org/51/4817

```
postgres=# create table t(a int);
postgres=# copy t from stdin(on_error table, table err_tbl);
Enter data to be copied followed by a newline.
End with a backslash and a period on a line by itself, or an EOF
signal.
>> a
>> \.
NOTICE:  1 row was saved to table "err_tbl" due to data type
incompatibility
COPY 0
postgres=# select * from err_tbl \gx
-[ RECORD 1 ]---+-----------------------------------------
userid          | 10
copy_tbl        | 18007
filename        | STDIN
lineno          | 1
line            | a
colname         | a
raw_field_value | a
err_message     | invalid input syntax for type integer: "a"
err_detail      |
errorcode       | 22P02
```

P.S. *err_tbl* is an existing table that's already set up for storing errors.
Collaboration wanted:

- Syntax `COPY ON_ERROR(table, err_tbl)` isn't ideal, but I haven't found a way to automatically save errors to a magic table without explicitly specifying it.
- Would appreciate a review—especially suggestions for improving the error messages.
- Currently, if a single field has an error, we skip to the next row. Should we instead capture all errors in a row and save them to the error saving table?

using indexscan to speedup add not null constraints to an table

jian he jian.universality@gmail.com

commitfest: https://commitfest.postgresql.org/patch/5444

By using index lookups instead of a full table scan to verify column's NOT NULL status, we can speed up the process of adding the NOT NULL constraint. If this is doable, then we can apply it to check constraints too.

```
create unlogged table t(a int, b int) with (autovacuum_enabled = off,
vacuum_index_cleanup=off);
insert into t select case when g % 2 = 0 then null else g end, g+1
from generate_series(1,1_000_000) g;
create index t_idx_a on t(a);
delete from t where a is null;
```

The table below shows the execution time (in milliseconds) for the above script adding a not-null constraint

HEAD	PATCH
8.621	0.423

If the column doesn't have any NULL values, using index lookup to add a not-null constraint would be way faster.
Collaboration wanted:

- Alter table adding a not-null constraint using will take an ACCESS EXCLUSIVE lock, which minimizes the chances of race (concurrency) issue. Could someone else take a look to confirm this?