POSTGRESQL is an open-source, full-featured relational database. This presentation gives an overview of the Postgres 15 release.
1. **MERGE**
2. **SQL/JSON** functions
3. Logical replication
4. Compression
5. Granular permissions
6. Memory
7. **COPY** headers

Part of the SQL standard, often requested

Similar to INSERT … ON CONFLICT, except

- join oriented, not row oriented
- does not require a unique index
- can error on concurrent changes
CREATE TABLE test (x INTEGER, y BOOLEAN);

INSERT INTO test VALUES (1), (3), (5);

INSERT INTO test VALUES (1), (2), (3), (4), (5), (6)
ON CONFLICT (x) DO UPDATE SET y = TRUE;
ERROR: there is no unique or exclusion constraint matching the ON CONFLICT specification

CREATE UNIQUE INDEX i_test ON test (x);

INSERT INTO test VALUES (1), (2), (3), (4), (5), (6)
ON CONFLICT (x) DO UPDATE SET y = TRUE;

SELECT * FROM test;
  x | y
---+----
  1 | t
  2 | (null)
  3 | t
  4 | (null)
  5 | t
  6 | (null)
DELETE FROM test;

INSERT INTO test VALUES (1), (3), (5);

MERGE INTO test
USING (VALUES (1), (2), (3), (4), (5), (6)) m (x)
ON test.x = m.x
WHEN NOT MATCHED THEN
    INSERT (x) VALUES (m.x)
WHEN MATCHED THEN
    UPDATE SET y = TRUE;

SELECT * FROM test;
  x  | y
---+---
  1  | t
  2  | (null)
  3  | t
  4  | (null)
  5  | t
  6  | (null)
These new functions provide an SQL-standard way to:

- create JSON scalars, objects, and arrays
- test the type of JSON value
- test the existence of JSON paths
- apply JSON paths
- convert a JSON document to a virtual table
3. Logical Replication

Logical replication is now more flexible by allowing:
- Publication of entire schemas, including future table additions
- Publication row control with a WHERE clause
- Publication column control
- Subscribers to skip specific transactions

Additional features are:
- Support for prepared transactions
- Suppress replication of empty transactions
- Possible replication termination on error
4. Compression

- Add LZ4 compression to the base backup protocol (gzip was already supported)
- Add LZ4 and Zstandard compression of pg_basebackup files (gzip was already supported)
- Allow pg_basebackup to control if compression happens server-side or client-side
- Add LZ4 compression to pg_receivelog (gzip was already supported)
- Add LZ4 and Zstandard compression of full page writes (LZ was already supported)
5. Granular Permissions

- Allow view to be run with the permissions of the view user, not owner
- Allow GRANT to control changes to server-side variables
- Add predefined role with checkpoint permission
6. Memory

- Improve performance of sorts that exceed work_mem
- Improve performance and reduce memory usage of in-memory sorts
- Store run-time server statistics in shared memory, rather than on disk
- Make hashing by default use twice as much memory as other operations
- Add server variable to report the amount of used shared memory and huge pages
CREATE TABLE copytest (x INTEGER, y TEXT);

INSERT INTO copytest VALUES (1, 'My term paper'), (2, 'Crossword puzzle');

COPY copytest to STDOUT;
1    My term paper
2    Crossword puzzle

COPY copytest to STDOUT WITH (HEADER);
 x    y
1    My term paper
2    Crossword puzzle

Previously only COPY’s CSV mode supported headers.
COPY copytest TO '/tmp/p' WITH (HEADER);

DELETE FROM copytest;

COPY copytest FROM '/tmp/p';
ERROR: invalid input syntax for type integer: "x"
CONTEXT: COPY copytest, line 1, column x: "x"

COPY copytest FROM '/tmp/p' WITH (HEADER);

SELECT * FROM copytest;
 x | y
---+------------------
 1 | My term paper
 2 | Crossword puzzle
ALTER TABLE copytest RENAME y TO z;

COPY copytest FROM '/tmp/p' WITH (HEADER);

COPY copytest FROM '/tmp/p' WITH (HEADER MATCH);
ERROR: column name mismatch in header line field 2: got "y", expected "z"  
CONTEXT: COPY copytest, line 1: "x     y"
Conclusion