
sqlalchemy-ibmi

Sep 09, 2021

Contents

1 DBAPI Connection	1
2 Connection Arguments	3
3 Transaction Isolation Level / Autocommit	5
4 Table Creation String Size	7
5 Query Function Strings	9
6 Text search support	11
Python Module Index	13
Index	15

CHAPTER 1

DBAPI Connection

This dialect uses the `pyODBC` DBAPI and the `IBM i Access ODBC Driver`.

Connection string:

```
engine = create_engine("ibmi://user:password@host/rdbname[?key=value&key=value...]")
```

Connection Arguments

The sqlalchemy-ibmi dialect supports multiple connection arguments that are passed in the URL to the `create_engine` function.

Connection string keywords:

- `current_schema` - Define the default schema to use for unqualified names.
- `library_list` - Specify which IBM i libraries to add to the server job's

library list. Can be specified in the URL as a comma separated list, or as a keyword argument to the `create_engine` function as a list of strings * `autocommit` - If `False`, `Connection.commit` must be called;

otherwise each statement is automatically committed. Defaults to `False`.

- `readonly` - If `True`, the connection is set to read-only. Defaults to `False`.
- `timeout` - The login timeout for the connection, in seconds.
- **`use_system_naming` - If `True`, the connection is set to use the `System` naming convention, otherwise it will use the `SQL` naming convention. Defaults to `False`.**
- **`trim_char_fields` - If `True`, all character fields will be returned with trailing spaces truncated. Defaults to `False`.**

create-engine arguments:

- `fast_executemany` - Enables PyODBC's `fast_executemany` option.

Conversion between input and target types is mostly unsupported when this feature is enabled. eg. Inserting a `Decimal` object into a `Float` column will produce the error "Converting decimal loses precision". Defaults to `False`.

Transaction Isolation Level / Autocommit

Db2 for i supports 5 isolation levels:

- SERIALIZABLE: *RR
- READ COMMITTED: *CS
- READ UNCOMMITTED: *CHG
- REPEATABLE READ: *ALL
- NO COMMIT: *NC

At this time, sqlalchemy-ibmi supports all of these isolation levels except NO COMMIT.

Autocommit is supported on all available isolation levels.

To set isolation level globally:

```
engine = create_engine("ibmi://user:pass@host/", isolation_level='REPEATABLE_READ')
```

To set using per-connection execution options:

```
connection = engine.connect()
connection = connection.execution_options(
    isolation_level="SERIALIZABLE"
)
```

Table Creation String Size

When creating a table with SQLAlchemy, Db2 for i requires that the size of a String column be provided.

Provide the length for a String column as follows:

```
class User(Base):
    __tablename__ = 'users'
    id = Column(Integer, Sequence('user_id_seq'), primary_key=True)
    name = Column(String(50))

users = Table('users', metadata,
              Column('id', Integer, Sequence('user_id_seq'), primary_key=True),
              Column('name', String(50)),
              )
```

Query Function Strings

Db2 for i doesn't support parameter markers in the SELECT clause of a statement. As a result, the following command will not work with sqlalchemy-ibmi:

```
session.query(func.count("*")).select_from(User).scalar()
```

Instead, please use the following:

```
session.query(func.count()).select_from(User).scalar()
```

Additionally, column names cannot be passed as strings, so convert your column strings to literal columns as follows:

```
from sqlalchemy.sql import literal_column
session.query(func.count(literal_column("colname"))).select_from(User).scalar()
```


CHAPTER 6

Text search support

The `ColumnOperators.match` function is implemented using a basic LIKE operation by default. However, when `Omnifind Text Search Server for Db2 for i` is installed, `match` will take advantage of the `CONTAINS` function that it provides.

S

sqlalchemy_ibmi.base, ??

S

sqlalchemy_ibmi.base (*module*), 1