Pemrograman Desktop 9

Yonathan F. Hendrawan

Relasi Tabel

- Materi sebelumnya hanya menampilkan data dari 1 tabel.
- Kita bisa mengakses data pada tabel lain dalam RDBMS menggunakan mekanisme primary key – foreign key
- Di PyQt, QSqlRelationalTableModel dapat digunakan untuk hal ini

Relasi Tabel

- Format perintah:

```
from PyQt6.QtSql import QSqlRelation, QSqlRelationalTableModel

self.model = QSqlRelationalTableModel(db=db)

relation = QSqlRelation('<related_table>',
'<related_table_foreign_key_column', '<column_to_display>')
self.model.setRelation(<column>, relation)
```

- Contoh:

```
self.model.setTable("Track")
self.model.setRelation(
    2, QSqlRelation("Album", "AlbumId", "Title")
)
self.model.setRelation(
    3, QSqlRelation("MediaType", "MediaTypeId", "Name")
)
self.model.setRelation(
    4, QSqlRelation("Genre", "GenreId", "Name")
)
self.model.select()
```

Relasi Tabel

```
class MainWindow(QMainWindow):
    def __init__(self):
        super().__init__()
        self.table = QTableView()
        self.model = QSqlRelationalTableModel(db=db)
        self.table.setModel(self.model)

        self.model.setTable("Track")
        self.model.setRelation(
            2, QSqlRelation("Album", "AlbumId", "Title")
        )
        self.model.setRelation(
            3, QSqlRelation("MediaType", "MediaTypeId", "Name")
        )
        self.model.setRelation(
            4, QSqlRelation("Genre", "GenreId", "Name")
        )
        self.model.select()

        self.setMinimumSize(QSize(1024, 600))
        self.setCentralWidget(self.table)
```

Edit Related Fields

- Contoh sebelumnya tidak dapat melakukan editing pada tabel relasi
- PyQt menyediakan QSqlRelationalDelegate yang dapat melakukan editing dengan menampilkan ComboBox

Edit Related Fields

```
class MainWindow(QMainWindow):
    def __init__(self):
        super().__init__()
        self.table = QTableView()
        self.model = QSqlRelationalTableModel(db=db)
        self.table.setModel(self.model)

        self.model.setTable("Track")
        self.model.setRelation(
            2, QSqlRelation("Album", "AlbumId", "Title")     )
        self.model.setRelation(
            3, QSqlRelation("MediaType", "MediaTypeId", "Name")     )
        self.model.setRelation(
            4, QSqlRelation("Genre", "GenreId", "Name")     )

        delegate = QSqlRelationalDelegate(self.table)
        self.table.setItemDelegate(delegate)

        self.model.select()
        self.setMinimumSize(QSize(1024, 600))
        self.setCentralWidget(self.table)
```

Query dengan SQL

- Kita dapat juga melakukan query menggunakan perintah SQL

Query dengan SQL

```
class MainWindow(QMainWindow):
    def __init__(self):
        super().__init__()
        self.table = QTableView()

        self.model = QSqlQueryModel()
        self.table.setModel(self.model)

        query = QSqlQuery("SELECT Name, Composer FROM track ", db=db)

        self.model.setQuery(query)

        self.setMinimumSize(QSize(1024, 600))
        self.setCentralWidget(self.table)
```

Query dengan SQL 2

- Contoh sebelumnya hanya melakukan perintah query sederhana
- Kita juga bisa melakukan query kompleks: antar beberapa tabel dan menggunakan filter

```
query = QSqlQuery(db=db)
query.prepare(
    "SELECT Name, Composer, Album.Title FROM Track "
    "INNER JOIN Album ON Track.AlbumId = Album.AlbumId "
    "WHERE Album.Title LIKE '%' || :album_title || '%' "
)
query.bindValue(":album_title", "Sinatra")
query.exec()
```

```
SELECT Name, Composer, Album.Title FROM Track
INNER JOIN Album ON Track.AlbumId = Album.AlbumId
WHERE Album.Title LIKE '%Sinatra%'
```

Query dengan SQL 2

```
class MainWindow(QMainWindow):
    def __init__(self):
        super().__init__()
        self.table = QTableView()
        self.model = QSqlQueryModel()
        self.table.setModel(self.model)

        query = QSqlQuery(db=db)
        query.prepare(
            "SELECT Name, Composer, Album.Title FROM Track "
            "INNER JOIN Album ON Track.AlbumId = Album.AlbumId "
            "WHERE Album.Title LIKE '%' || :album_title || '%' "
        )
        query.bindValue(":album_title", "Sinatra")
        query.exec()

        self.model.setQuery(query)
        self.setMinimumSize(QSize(1024, 600))
        self.setCentralWidget(self.table)
```