09

SQLite primarykey

Transcrição

[00:00] Agora que conseguimos visualizar os dados do nosso banco de dados SQLite, que foi usado para gravar os dados locais do nosso agendamento, analisando a estrutura, o esquema da tabela agendamento percebemos aqui um problema muito sério, porque essa tabela não tem a chave primária.

[00:26] Se ela não tem a chave primária deixamos de utilizar um dos grandes benefícios de um banco de dados relacional, que é o índice, que é a identificação de cada registro de uma tabela, agora porque essa tabela, porque esse esquema do banco SQLite foi criado sem uma chave primária, sem um índice, um identificador para cada registro dessa tabela?

[00:53] É muito simples, é porque não definimos isso no nosso modelo, como o banco SQLite, como a tabela de agendamento SQLite está sendo criado em cima do nosso modelo e o nosso modelo também não definimos um ID, não definimos uma chave primária, o esquema da tabela do banco de dados SQLite também vai ser criado sem esse identificador, sem essa chave primária.

[01:18] O que precisamos fazer é criar esse esquema com o Id no nosso modelo, na nossa classe agendamento que está lá no arquivo "Agendamento.cs", vamos modificar aqui a estrutura do nosso modelo, para incluir uma nova propriedade chamada "id", essa propriedade vai se transformar lá na frente em uma chave primária, que eu vou chamar de "ID".

[01:48] Agora não basta só fazer isso, o que eu preciso também é marcar essa propriedade "ID" como chave primária, como identificador da minha tabela de agendamento, como eu faço isso?

[02:02] Eu utilizo um atributo do SQLite chamado PrimaryKey, eu vou abrir aqui a chave para colocar o atributo e eu coloco "PrimaryKey", com isso eu já estou marcando o "ID" como chave primária, agora, só isso, será que basta?

[02:21] Se eu só faço isso e vou começar a inserir registros nessa tabela de agendamento, o que acontece? Eu vou ter que informar a chave, além de todos os dados que já estou passando para o agendamento, eu tenho que falar esse agendamento é o número 1, esse é o número 2, esse número 3, esse número 4, eu tenho que consultar a tabela primeiro para pegar o ID e verificar qual vai ser o próximo ID para poder inserir essa informação corretamente no banco.

[02:52] Mas eu não quero fazer isso, porque é um trabalho muito chato e muito redundante, eu estou o tempo todo tendo que recalcular qual é o Id que eu tenho que inserir na tabela, para evitar isso eu posso marcar a propriedade "ID" como um auto incremento, porque automaticamente a cada "insert" ele vai se atualizar, ele vai verificar qual é a próxima ID e vai fazer isso automaticamente para mim.

[03:20] Com isso eu não preciso me preocupar com a propriedade "ID' de forma nenhuma, eu simplesmente marco a propriedade "ID" como "AutoIncrement", eu vou ter uma numeração automática, não importa qual é o valor que eu estou passando para os outros registros, eu não preciso informar qual é o valor do ID, porque isso vai ser automaticamente criado pelo mecanismo do SQLite.

[03:45] Agora vamos rodar a aplicação e ver essa mudança acontecendo lá no banco de dados. Eu vou entrar com usuário "joao@alura.com.br", a senha "alura123", vou escolher um veículo, Sentra 2.0, próximo, vou agendar e vou confirmar, agora está dando uma exceção, não pode ser adicionada uma coluna chave primária, porque está acontecendo isso?

[04:17] O motivo é muito simples, eu já tenho uma tabela, já tenho um banco de dados SQLite, gravado no SD Card do meu emulador, quando eu estou tentando me conectar novamente e verificar, quando ele passa, quando o SQLite verifica aqui nessa linha, nessa linha exatamente eu tenho “this.conexão.CreateTable”, ele vai criar a tabela agendamento se ela não existir.

[04:47] Só que além disso ela faz uma outra coisa, ela verifica se o meu modelo, se a minha classe de modelo agendamento é compatível com banco SQLite, se a tabela que já está lá no esquema que já está no banco SQLite é compatível com o meu modelo, se for diferente ele vai reclamar e vai lançar uma exceção como ele acabou de fazer.

[05:11] O que precisamos fazer para consertar isso é remover o banco de dados SQLite que já foi gravado lá no nosso emulador, vamos remover esse banco de dados “Agendamento.db3” e vamos rodar de novo, porque ele vai recriar e vai poder utilizar o novo esquema contendo a chave primária.

[05:35] Vou selecionar o arquivo “db3”, e agora eu vou clicar aqui na lixeira, aqui em cima e vamos remover esse arquivo, pronto, já acabei de remover e vou rodar de novo a aplicação, eu vou entrar com “joao@alura.com.br”, senha “alura123”, e vou escolher um veículo, vou escolher C3 1.0, clicar no próximo, agendar e vou confirmar o agendamento, agora vai passar aqui no “Insert”, inseriu sem problemas.

[06:11] Conseguimos trocar o esquema, trocar a estrutura da tabela de agendamento para poder ter agora a coluna chave primária, que é a coluna ID, que vai ser um auto incremento também.

[06:28] Agora depois que fizemos isso, queremos ver os dados gravados nessa tabela, queremos ver a nova estrutura dela gravada, como fazemos?

[06:37] Vamos simplesmente rodar de novo o nosso comando “adb pull”, que vai trazer lá da pasta de SDCard do emulador para a nossa máquina local, para máquina de desenvolvimento, eu vou rodar aqui de novo, está fazendo a cópia, copiou os arquivos, agora eu vou abrir novamente, vou fechar o que eu já tinha e agora vou abrir novamente o arquivo “Agendamento.db3”.

[07:06] Agora vamos abrir a estrutura da tabela agendamento, agora temos o ID que é uma nova propriedade, que também é um inteiro e é uma chave primária com auto incremento, ID inteiro “not null”, chave primaria com auto incremento.

[07:30] Se vamos navegar os dados agora, eu tenho aqui o “ID”, com o valor “1”, em nenhum momento eu passei o valor “1” para propriedade “ID”, só que o SQLite sabe que essa propriedade é um auto incremento e é uma chave primaria, então automaticamente ele criou para mim com outra numeração, se inserirmos mais um registro vamos inserir o “2”, o “3” e assim por diante. Para finalizar vamos dar uma olhada aqui na nossa navegação e vamos corrigir um problema que está acontecendo.

[08:03] Quando eu agendo, acabei de agendar um TestDrive, ele não sai dessa tela, se eu clico de novo em agendar ele salva o agendamento, salvou, inseriu o agendamento e deu uma falha, se eu tentar de novo ele deu um sucesso. Só que ele não sai dessa tela, eu não quero que aconteça isso, o que eu quero é que voltemos lá para o começo, onde já tem os veículos prontos para o meu agendamento, eu quero voltar para tela inicial, para listagem de veículos e por que eu quero isso?

[08:44] Porque eu quero poder agendar novos veículos, novos TestDrive de veículos sem ter que voltar aqui com o botão “voltar”, sem ter que clicar aqui em cima, estou voltando, eu agendo um Uno, eu vou para frente, depois eu salvo, eu tenho que voltar, quero voltar automaticamente para o ponto de partida que é a listagem de veículos, como eu faço isso?

[09:07] Aqui na tela de agendamento, quando o agendamento for enviado para o servidor eu vou ter que enviar uma mensagem para fazer com que a navegação volte para início, para raiz.

[09:22] Vamos aqui no AgendamentoView.xaml.cs, que é o Code behind da View do agendamento e vamos modificar, fazer uma mudança aqui, para que tanto no sucesso quanto na falha voltemos para o início da pilha de navegação e como fazemos isso?

[09:42] Vamos ter que utilizar o serviço de navegação, vamos ter que utilizar a classe Navigation do Xamarin Forms, logo após o sucesso agendamento vamos ter que passar aqui "Navigation" e vamos ter que chamar um método que vai fazer essa navegação de volta para o início da pilha, chamamos um método que é para desempilhar. Se estamos empilhando com o push, que é em inglês de empurrar, vamos ter que puxar, vamos ter que desempilhar com o comando "pop".

[10:16] Usamos o "pop", para onde queremos puxar? Queremos puxar até chegar na raiz, "PopToRoot" e o "Async" dizendo que é um método assíncrono, usamos esse comando para poder navegar de volta até a página inicial, até a página raiz da nossa navegação, além disso eu também tenho que marcar esse comando aqui em cima "DisplayAlert" com o operador "await", então eu vou colocar "await" para aguardar a resposta do usuário, para o "ok" do usuário e eu vou chegar no "PopToRootAsync" para poder navegar para o início.

[11:00] Eu vou marcar essas duas linhas com "await" e vou fazer a mesma coisa aqui embaixo para falha, quando tiver falha do agendamento eu quero também que ele vá lá para o início porque vamos na próxima aula ver como vamos salvar o agendamento mesmo com falha, vamos fazer uma representativa de salvar agendamentos que falharam.

[11:23] Marcarmos aqui o "DisplayAlert" com "await" e vamos colocar uma nova linha aqui embaixo para fazer essa navegação para a raiz da pilha de navegação, "await Navigation.PopToRootAsync".

[11:45] Agora o VisualStudio está reclamando porque estamos utilizando "await" dentro de um método que não está marcado como "async", então vamos corrigir isso marcando aqui essa expressão lambda, esse método anônimo "async", eu coloco "async" aqui, resolvido, aqui em baixo também, eu marco como "async", resolveu esse problema do Visual Studio.

[12:09] Feito isso, o nosso método "OnAppearing" está muito grande, para deixar ele mais compacto o que eu vou fazer? Eu vou extrair um método que vai fazer com que essas assinaturas de mensagem fiquem em um método específico para isso, eu vou extrair o método aqui com "Quick Actions" e vou escolher, extrair método, vou clicar em extrair método e vou dar o nome de "AssinarMensagens".

[12:44] Agora vamos rodar a aplicação de novo e ver se a navegação está correta agora, vou entrar com usuário "joao@alura.com.br", senha "alura123", vou entrar agora, vou escolher um carro Astra Sedan, clicar no próximo, vou agendar, confirmar agendamento, falha ao agendar, dou "Ok" e ele voltou para o início, voltou para listagem Inicial, vou clicar no outro veículo aqui OutLander, escolher próximo, agendar, confirmar, agendou com sucesso, salvou o agendamento, clico no "Ok" e ele volta lá para o início para listagem de veículos.

[13:26] Com isso terminamos a aula de gravação de banco de dados, utilizando Xamarin Forms com o banco de dados SQLite, espero que vocês tenham gostado, muito obrigado, até a próxima.